Departamento de Enxeñaría de Computadores
Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN ENXEÑERÍA INFORMÁTICA
MENCIÓN EN ENXEÑARÍA DE COMPUTADORES

Dispositivo e aplicación Android para aumentar a seguridade nos desprazamentos en bicicleta

Estudante: Sergio Rodríguez Gayoso
Director/a/es/as: Carlos Vázquez Regueiro

A Coruña, 6 de setembro de 2019.

Agradecementos

A meus pais polo seu apoio, a meus profesores e director de proxecto pola sua dedicación e paciencia e aos meus colegas Martín e Carlos pola sua axuda coas probas e vídeos.

Resumo

Hoxe en día os desprazamentos en bicicleta, especialmente en cidade, están a sufrir un notable incremento. A concienciación ecoloxista e a saturación de coches que sofre o entorno urbano esta levando á xente a buscar métodos de desprazamentos unipersoais. A convivencia destes vehículos cos coches implica un alto risco de lesións en caso de accidente. Para tratar de mellorar esta situación buscarase crear un sistema para aumentar a visibilidade do usuario como a súa capacidade de visión.

Analizaremos as opcións posibles, e deseñaremos e implementaremos un sistema baseado en dous dispositivo BikeCam, un miniordenador cunha cámara e luces LED de cores que usaremos para indicar a posición e as manobras do usuario, e BikeView, unha aplicación Android que permitirá a visualización do vídeo en directo así como o control automatizado das luces.

Palabras chave:

- Seguridade
- Bicicleta
- Luces
- Cámara
- Vídeo
- Batería
- Rasberry Pi
- Android

Índice Xeral

1 Introdución **1**

1.1 Motivación . . . 1
1.2 Obxectivos . . . 2
1.3 Proposta . . . 3
1.4 Traballo relacionado . . . 3
1.5 Estrutura da memoria . . . 4

2 Analise e planificación **5**

2.1 Metodoloxía . . . 5
2.2 Análise . . . 5
2.3 Planificación . . . 7
2.3.1 Fase 1: Dispositivo principal . . . 7
2.3.2 Fase 2: Aplicación do dispositivo móbil . . . 8
2.4 Custos do proxecto . . . 8

3 Arquitectura do sistema BikeGuard **11**

3.1 Arquitectura do sistema . . . 11
3.2 Esquema xeral de BikeView . . . 11
3.3 Esquema xeral de BikeCam . . . 12
3.3.1 Placas de desenvolvemento . . . 13
3.3.2 Luces LED . . . 16
3.3.3 Cámara . . . 17
3.3.4 Alimentación e baterías . . . 18
3.3.5 Software a utilizar . . . 19
3.3.6 Caixa e ancoraxes á bicicleta . . . 20
3.3.7 Esquema conceptual de BikeCam . . . 20
3.3.8 Esquema conceptual de BikeLed . . . 20
3.4 Comunicacións entre BikeCam e BikeView . . . 20

3.4.1 Canle de comunicaións . . . 22
3.4.2 Protocolos de comunicación . . . 23
3.4.3 Protocolos de videostreaming . . . 24

4 Deseño e implementación de BikeCam 25
4.1 Estrutura xeral de BikeCam . . . 25
4.2 LEDs . . . 26
4.2.1 Conexión coa Raspberry Pi . . . 26
4.3 Cámara . . . 27
4.3.1 Lentes . . . 28
4.4 Alimentación e enerxía . . . 28
4.5 Software de control de BikeCam . . . 34
4.5.1 lightsServer . . . 34
4.5.2 Captura de vídeo . . . 36
4.5.3 Transmisión de vídeo . . . 36
4.5.4 LipoPi . . . 37
4.5.5 Servizos en *systemd* . . . 37
4.6 Carcasa e ancoraxe . . . 38
4.6.1 Versión con batería externa . . . 38
4.6.2 Versión con batería interna . . . 38

5 Deseño e implementación de BikeView 41
5.1 Esquema xeral da aplicación . . . 41
5.2 Actividade principal . . . 42
5.2.1 Layout . . . 42
5.2.2 Ciclo de vida da actividade . . . 42
5.2.3 Botons . . . 44
5.2.4 Sensores . . . 45
5.2.5 Superficie de vídeo . . . 47
5.3 Comuniciacíon co dispositivo . . . 47
5.3.1 Broadcast . . . 47
5.3.2 Conexión . . . 47
5.3.3 Transmisión de ordes . . . 47
5.4 Recepción de vídeo . . . 48
5.5 Ancoraxe á bicicleta . . . 48

6 Evaluación experimental **51**
6.1 Consumo . . . 51
6.2 Autonomía . . . 53
6.3 Vídeo e lentes . . . 54
6.4 Visibilidade . . . 55
6.4.1 Modo noite . . . 57
6.4.2 Freo automático . . . 57
6.5 Lexislación . . . 57
6.5.1 Luces . . . 57
6.5.2 Cámaras . . . 59

7 Conclusións e traballo futuro **61**
7.1 Conclusións . . . 61
7.2 Traballo futuro . . . 62

A Apéndices **63**

B Contido DVD **65**

C Requisitos e instalación **67**

D Glosario de acrónimos **69**

E Glosario de termos **71**

Bibliografía **73**

Índice de Figuras

3.1 Esquema xeral do prototipo BikeGuard desenvolto neste proxecto. 12
3.2 Esquema do aspecto da aplicación BikeView. 13
3.3 Exemplo de implementación de BikeCam. 21
3.4 Exemplo de implementación de BikeLed. 21

4.1 Diagrama de conexión dos LEDs. 27
4.2 De esquerda a dereita a Raspberry Pi Camera Module V1, V2 sen filtro infravermello e V2 cunha lente de gran angular 28
4.3 Lentes con diferente ángulo de visión. De esquerda a dereita: fila superior 0.67x, 180º, 0.4x, fila inferior 130º, 160º, 220º 29
4.4 Diagrama de funcionamento do circuíto de alimentación. 31
4.5 Diagrama do dispositivo. 32
4.6 Esquema completo do dispositivo. 33
4.7 Imaxes do circuíto por ambas caras. 34
4.8 Deseños do soporte na bicicleta para a caixa oficia da Rasberry Pi. 39
4.9 Ultimo deseño da carcasa. 40

5.1 Capturas de pantalla do *layout* horizontal. 43
5.2 Capturas de pantalla do *layout* vertical. 43
5.3 Eixos do acelerómetro e ángulo coa horizontal. Para o dispositivo colocado en horizontal, en vertical o eixo superior sería y. 46
5.4 Mostras dos problemas do soporte impreso. 49

6.1 Grafica do consumo en mA. 52
6.2 Capturas feitas coa cámara V1, V2, e V2 sen filtro infravermello. 54
6.3 Comparativa de lentes de 130º 180º e 220º 55
6.4 Perecepción subxectiva de estímulos lumínicos segundo a lei de Weber-Fechner [1]. 56
6.5 Fotos (recortadas) da luz vermella a 150m. A esquerda de día a dereita de noite. 56

6.6 Comparación das luces vermellas xunto as luces de freada dun coche a 150m. . 56
6.7 Rango de intensidade lumínica nos faros das biciletas en España [2]. 58

Índice de Táboas

1.1 Lesividade en función da luminosidade da vía [3]. 2

2.1 Custos monetarios do proxecto 9
2.2 Custos de fabricación do dispositivo *DIY* 9
2.3 Tempo empregado no proxecto 10

4.1 Táboa comparativa da Raspberry Pi Camera V1 e V2 [4] 28

6.1 Características dos LEDs WS2812B de Worldsemi 59

Capítulo 1

Introdución

Neste capítulo introdutorio preséntase a motivación e os obxectivos do presente traballo. Por último, comentarase a estrutura da memoria.

1.1 Motivación

Nos últimos anos os medios de transporte alternativos como son o caso de bicicletas, patinetes e similares están aumentando notablemente. As preocupacións medioambientais, os beneficios para a saúde e as vantaxes en termos de custo e eficiencia están a impulsar estes vehículos unipersoais. Porén, a falta de costume dos condutores xunto coas substanciais diferencias entre os vehículos e a falta de protección en caso de accidente dificultan a convivencia con automóbiles nas mesmas vías e implican un risco engadido para a integridade física dos condutores destes vehículos. Unha das maneiras de reducir estes riscos é aumentar a visibilidade tanto da bicicleta por parte dos coches coma viceversa co obxectivo de aumentar as distancias e os tempos de reacción dos condutores.

A maioría dos accidentes nos que a vítima é un ciclista implican un turismo na colisión [5]. O estudo da Universidade de Valencia [3] con datos da Dirección Xeneral de Tráfico mostra que a pesar de que a maioría de accidentes ciclistas prodúcense de día e con boas condicións de visibilidade, momento no que hay maior número desprazamentos en bicicleta, de noite ou con pouca luminosidade a gravidade das lesións é superior, como se mostra na táboa 1.1.

Varios estudos mostran que o uso de luces en bicicletas, especialmente as dinámicas permiten que estas sexan divisadas a maiores distancias tanto de día como de noite. A tese The Nighttime Conspicuity Benefits of Static and Dynamic Bicycle Taillights [6] estuda os beneficios de diferentes luces traseiras de bicicleta de noite. Conclúe que as luces que se moven co ciclista, como as colocadas nos nocellos, son as que máis visibilidade aportan pero cando o ciclista deixa de pedalear estes beneficios pérdense polo que o uso dunha luz fixa cun patrón de intermitencia pode ser a mellor opción en tódalas circunstancias.

Táboa 1.1: Lesividade en función da luminosidade da vía [3].

	Morto	Ferido grave	Ferido leve l
Pleno día	1.1%	13.7%	85.1%
Crepúsculo	1.9%	13.0%	85.1%
Iluminación suficiente (noite)	0.7%	9.3%	90.0%
Iluminación insuficiente (noite)	2.8%	17.7%	79.5%
Sen iluminación (noite)	10.3%	27.4%	62.4%

Por outra parte a falta de retrovisores na maioría de bicicletas implica que o ciclista debe xirarse cada vez que quere saber o que está a acontecer tras el facendo que así perda momentaneamente a visión do que ocorre diante e incluso o equilibrio en ciclistas non experimentados.

Para paliar estes problemas exponse unha solución baseada nun ou varios dispositivos dotados de luces e cámara xunto a outro de control e visualización.

1.2 Obxectivos

Os obxectivos principais de este proxecto serán dous: O primeiro de desenvolvemento de dous dispositivos diferenciados, un dotado de cámara e luces que se poderá colocar en diferentes lugares da bicicleta e do que se poderán utilizar unha ou varias unidades ao mesmo tempo que disporá de alimentación propia ou compartida. Un segundo dispositivo de interacción co usuario para o manexo do primeiro e a visualización do vídeo capturado. Tamén será necesario o hardware que permita a comunicación dos dispositivos xa sexa por cable ou sen fíos.

O segundo consistirá en desenvolver o software que permita o funcionamento dos dispositivos. O control das luces para indicar a posición, aumentar a visibilidade ou indicar manobras coma a freada ou xiro. O control do vídeo permitindo activalo e desactivalo cando se desexe. As posibles automatizacións como o acendido de luces cando haia pouca visibilidade ou a indicación automática da freada. As interfaces de iteración co usuario que permitan o control e a visualización sen distraccións da condución. A integración e comunicación entre os dispositivos en tempo real e de forma transparente para o usuario.

Comenzarase coa análise das posíbeis solucións contemplando as diferentes opcións de hardware dispoñibles tendo en conta o custo, o tamaño, as capacidades de funcionamento, as restricións de compatibilidade, as restricións no software a utilizar, a dispoñibilidade e a dificultade de uso polo usuario final.

Realizarase a implementación da solución elixida e someterase a probas nun entorno real para comprobar o cumprimento dos requisitos establecidos.

1.3 Proposta

Neste traballo propoñerase unha solución baseada nun sistema con dous tipos de compoñentes interconectados, uns encargados de comunicarse co medio e outros de comunicarse co usuario.

Dentro do primeiro tipo situarase o dispositivo a prototipar neste proxecto, un microordenador colocado baixo a sela, a Raspberry Pi Zero, que contará cunha cámara é unhas serie de luces LEDs conectadas, outro exemplo sería un dispositivo máis sinxelo que só contara con luces e fora controlado por un Arduino ou similar, este podería substituír o dispositivo baixo a sela ou complementalo facendo de luces frontais ou laterais.

No segundo tipo situaremos a aplicación Android que desenvolveremos neste proxecto, encargada do control e da visualización do vídeo en directo, que se executará nun teléfono situado no guiador da bicicleta.

Para a interconexión entre dispositivos contémplanse varias opción dende USB para conseguir a máxima tas de transmisión de datos e mínima latencia ata Bluetooth para contar cun baixo consumo nos dispositvos máis sinxelos. O sistema da solución prototipada comunicarase por Wi-Fi o que permitirá unha conexión rápida entre os dispositivo o tempo que estes manteñen a independencia.

1.4 Traballo relacionado

Non existen no mercado dispositivos con funcionalidades similares, os dous únicos que contan con algunha das característica buscadas por este proxecto son os seguintes.

A cámara Fly6 da compañía Cycliq [7], un combo de cámara traseira máis luces que se poden controlar dende o móbil, pero non permite o *streaming* en directo do vídeo, solo a gravación. Conta con características interesantes como a estabilización da imaxe, resistencia á auga e un tamaño moi compacto, o seu prezo é de 179 euros.

A outra opción existente é a cámara Hexagon [8] que se financiou exitosamente mediante croudfounding no ano 2017 pero nunca chegou a produción. Este dispositivo si que contaría con *streaming* de vídeo en directo, xunto con acendido automático das luces en caso de freada. A aplicación tamén se encarga de gravar a posición gps no itinerario ou compartila en tempo real, conta cun segundo dispositivo con botóns para o control do dispositivo, e de chegar a producirse o seu prezo estaría entre os 100 e 200 euros.

A principal diferenza destes dispositivos con este proxecto é o uso dunha arquitectura aberta e modular que non so integra máis funcionalidades se non que sentará como base para a implantación de moitas máis no futuro.

1.5 Estrutura da memoria

Este documento estrutúrase en sete capítulos e dous anexos.

- Neste primeiro capítulo exponse a motivación os requisitos e as liñas xerais do proxecto, incluíndo os obxectivos e finalmente faise unha proposta e compárase con traballos relacionados.
- No capitulo 2 expóñense as funcionalidades requiridas para o sistema, elíxese a metodoloxía a utilizar e partindo de esta planifícanse as etapas de desenvolvemento e por ultimo móstranse os custo do proxecto.
- O capítulo 3 presenta a arquitectura do sistema e explora as posibles alternativas dispoñibles analizando as vantaxes e desvantaxes de cada unha delas fronte a solución elixida.
- O capítulo 4 relata o proceso, os detalles e os problemas xurdidos na implementación e construción do dispositivo BikeCam xunto co seu software.
- O capítulo 5 expón o deseño e implementación de BikeView a aplicación Android que se utilizará para o control e a visualización.
- O capítulo 6 describe as probas realizadas e a análise dos seus resultados. O sexto e derradeiro capítulo recolle as conclusión obtidas trala realización deste proxecto.
- O capítulo 7 é o derradeiro no que se presentan as conclusións finais do proxecto valorando o proceso levado a cabo e os resultados obtidos, finalmente se formulan opcións de traballo futuro e posibles melloras do proxecto.

Capítulo 2

Analise e planificación

Neste capítulo expoñerase a metodoloxía a utilizar, partirase dos requisitos do proxecto para analizar as posibles funcionalidades e planificarase o proceso de implementación. Por último preséntanse os custos do proxecto.

2.1 Metodoloxía

A forma de traballo consistirá en seguir as directrices dadas polo método da enxeñaría partindo de estudo de traballos relacionados anteriores e continuando co análise, deseño, implementación e avaliación do sistema implementado.

No apartado de deseño, debido a que o proxecto se estrutura en diferentes compoñentes, optarase por unha metodoloxía Top-Down [9] que consiste en, partindo dos requisitos xerais, dividir o sistema en módulos independentes que se implementarán e probarán por separado. A cada un destes módulos se lle aplicará unha metodoloxía iterativa baseada en prototipos engadindo en cada incremento novas funcionalidades ao prototipo previamente desenvolvido, implementado e validado.

2.2 Análise

O obxetivo do proxecto é a creación de dous dispositivos que realizarán as seguintes funcións:

- **Informar da posición da bicicleta mediante luces**

  Esta función realizarase no dispositivo principal. Para elo terá que contar con luces LEDs e capacidade para controlalas, tamén dispor de capacidade de comunicación co segundo dispositivo e dispoñer dunha fonte de alimentación con capacidade suficiente para facer funcionar as luces.

- **Informar das manobras e estado do vehículo mediante luces**

  Para realizar esta función os requisitos son os mesmos da función anterior ao que engadiremos a necesidade de sensores para detectar cambios no movemento da bicicleta, para sinalizar freadas ou accidentes e cambios na luz do ambiente para acender as luces cando as condicións lumínicas non sexan favorables.

- **Captura de vídeo do que sucede detrás do vehículo**

  Esta función tamén se realizará no dispositivo principal e necesitará os requisitos de comunicación e alimentación enerxética xa citados. Contará cunha cámara para capturar as imaxes e necesitará a capacidade para procesalas e transmitilas en tempo real.

- **Entrada de ordes do usuario para o control de luces e vídeo**

  Esta función se executa no segundo dispositivo, para realizala será necesario un mecanismo de entrada, como poden ser botóns, pulsadores ou pantalla táctil. Tamén se necesitará capacidades de procesamento, comunicación e alimentación enerxética.

- **Reprodución do vídeo en tempo real**

  Tamén a realizar no dispositivo dous, esta función necesita, a maiores do anteriormente citado unha pantalla na que poder ver o vídeo, e capacidades abondo para reproducilo a tempo real.

A maiores destes requisitos funcionais contase cos seguintes obxectivos:

- **Pequeno tamaño e portabilidade**

  Co motivo de poder dispoñer o dispositivo da bicicleta, xa sexa para cargalo ou por motivos de seguridade o deixar a bicicleta aparcada na rúa, priorizarase por un deseño portátil. O ideal é que os dispositivos poidan separarse da bicicleta con facilidade e que o seu tamaño permita gardalos no peto. Este, xunto cos requisitos funcionais, é o principal motivo polo que optarase por utilizar un telefono móbil como segundo dispositivo, xa que a maioría de persoas dispoñen dun en todo momento, e así evitaríase ter que levar un dispositivo a maiores.

- **Independencia entre dispositivos**

  Buscarase que os dispositivos poidan conectarse sen fíos para obter unha maior independencia entre os dispositivos e evitar ter que colocar cables na bicicleta. Tamén se estudará a posibilidade de utilizar unha conexión cableada para diminuír o consumo

enerxético do dispositivo, neste caso optarase preferiblemente por unha conexión USB por compatibilidade co telefono móbil.

- **Batería e alimentación**

  Para poder alimentar o dispositivo principal necesitarase dunha batería con capacidade abonda para poder utilizalo polo menos un dia de uso sen ter que recargala. Tamén se estudará a posibilidade de incluír sistemas de aceso e apagado de ser preciso.

- **Sinxeleza e capacidade de actualización**

  Pretendese desenvolver un sistema robusto e simple para facilitar o seu mantemento e poder actualizalo de forma sinxela. Especialmente o software debe ser o máis simple posible para poder permitir incorporar novas funcionalidades no futuro.

2.3 Planificación

Seguindo a metodoloxía Top-Dow o desenvolvemento do proxecto dividirase en dúas fases correspondentes ós dous dispositivos. Comezarase polo desenvolvemento do dispositivo principal seguido da aplicación no dispositivo móbil é a comunicación entre ambos. Unha vez desenvolvidas as funcións principais iterarase entre as fase para desenrolar a conexión entre os dispositivos.

2.3.1 Fase 1: Dispositivo principal

As tarefas a realizar son as seguintes:

- Conexión dos LEDs e probas de funcionamento.
- Funcións de control dos LEDs, secuencias e cores.
- Servidor de peticións de ordes.
- Conexión da cámara e probas de funcionamento.
- Transmisión de vídeo.
- Autoarranque e apagado.
- Alimentación enerxética e batería.
- Deseño e construción de carcasa e ancoraxes.

2.3.2 Fase 2: Aplicación do dispositivo móbil

Realizaranse a seguintes tarefas:

- Introdución de ordes.
- Deseño de interfaces.
- Transmisión de ordes.
- Xestión do estado.
- Probas con sensores.
- Funcións de automatización.
- Recepción de vídeo.
- Ancoraxe do dispositivo á bicicleta.

2.4 Custos do proxecto

O software utilizado neste proxecto conta con licenzas de software libre polo que os custos restrinxiranse ós recursos humanos e recursos hardware. Nos recursos hardware da táboa 2.4 inclúense as pezas utilizadas finalmente para o dispositivo asi como as que se utilizaron para realización de probas. Para aproximar o rango prezos do custo *DIY* de fabricación do dispositivo na táboa 2.4 calcularanse dúas opción a primeira utilizando os compoñentes máis baratos é a segunda os máis caros.

As estes custos engádense as ferramentas utilizadas incluíndo entre outros ordenador, impresora 3D, multímetro, soldador, dispositivo android e bicicleta.

Na táboa 2.4 móstranse os recursos humanos utilizados medidos en horas de traballo.

Táboa 2.1: Custos monetarios do proxecto

Compoñente	Cantidade	Custo por unidade	Subtotal
Raspberry Pi Zero W	2	11.00€	22.00€
Raspberry Pi Zero	2	5.00€	5.00€
Caixa oficial Pi Zero	1	6.00€	6.00€
Pi Camera Module V1	1	5.00€	5.00€
Pi Camera Module V2	2	20.00€	40.00€
Conxunto de lentes	1	2.00€	2.00€
Tira 8 LEDs w2812b	4	1.00€	4.00€
Anel 8 LEDs w2812b	2	1.00€	2.00€
Adafruit Powerboost 1000C	1	23.00€	23.00€
Batería Lipo 1600mA	1	7.00€	7.00€
Batería 18650 3400mA	2	2.00€	4.00€
Material impresión 3D PLA	<1Kg	15.00€	15.00€
Cables			5.00€
Total			140.00€

Táboa 2.2: Custos de fabricación do dispositivo *DIY*

Menor prezo	Custo	Maior Prezo	Custo
Raspberry Pi Zero	5.00€	Raspberry Pi Zero W	11.00€
Adaptador Wi-Fi usb	1.00€		
Material Carcasa 3D	0.50€	Carcasa ofical	6.00€
		Material Soprte 3D	0.50€
Pi Camera Module V1	3.00€	Pi Camera Module V2	20.00€
		Lentes	2.00€
8 LEDs w2812b	1.00€	24 LEDs w2812b	3.00€
Circuito de carga e protección xenérico	2.00€	Adafruit Powerboost 1000C	23.00€
Batería Lipo 1600mA	7.00€	Batería Lipo 1800mA	10.00€
Cables	1.00€	Cables	1.00€
Total	20.50€	Total	76.50€

Táboa 2.3: Tempo empregado no proxecto

Tarefa	Tempo
Documentación	80h
Desenvolvemento do dispositivo	100h
Desenvolvemento do software do dispositivo	100h
Desenvolvemento da aplicación	200h
Deseño das pezas 3D	70h
Probas e avaliación experimental	32h
Total	512h

Capítulo 3

Arquitectura do sistema BikeGuard

Neste capítulo esbózase a arquitectura do sistema, baseándonos nos requisitos estúdanse as posibles opcións tecnolóxicas para a implementación de cada un dos dispositivos e do seu esquema de comunicación.

3.1 Arquitectura do sistema

BikeGuard será o nome que asignaremos ao sistema e se comporá de dous elementos principais BikeView, o dispositivo de control e visualización e BikeCam o dispositivo de captura de imaxes e sinalización lumínica, plantexarase unha variación de este, BikeLed, que so contará con elementos de iluminación, figura 3.1.

Neste sistema os dispositivos BikeCam e BikeLed terán a función de servidores e proporcionarán dous servizos ao cliente BikeView, encender e apagar as luces e patróns lumínicos, e proporcionarlle o vídeo en directo. Como todas as ordes partirán do dispositivo BikeView, tanto as do usuario como as automáticas a partir dos datos dos sensores do dispositivo Android, e para simplificar as conexión e manter a coerencia do estado do sistema este manterase no dispositivo BikeView.

Nas seguintes seccións analizaremos as alternativas para o desenvolvemento dos dous dispositivos e as conexións entre ambos.

3.2 Esquema xeral de BikeView

Son moitas as posibilidades de implementación deste dispositivo, os requisitos principais son:

- Dispoñer dunha pantalla para visualizar o vídeo e o estado das luces.
- Contar con algunha interface de entrada de datos como botóns ou pantalla táctil.

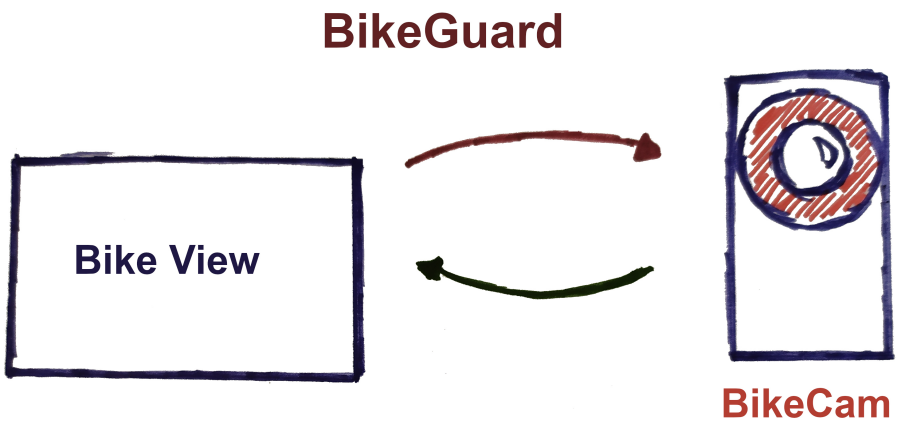

Figura 3.1: Esquema xeral do prototipo BikeGuard desenvolto neste proxecto.

- Dispoñer dun hardware para comunicarse co dispositivo principal. Como por exemplo: Wi-Fi, Bluetooth ou USB.
- Contar cunha batería ou fonte de alimentación

Realizar unha implementación física do dispositivo contaría coas vantaxes de poder contar cunha alta personalización dos seus compoñentes e robustez ao contar cun único hardware obxectivo, unha destas opción sería utilizar un miniordenador como a Rasperry Pi conectado a unha pantalla táctil. Sen embargo optarase outra opción moito máis atractiva: utilizar un telefono móbil e crear unha aplicación dende onde poder visualizar o vídeo e controlar as luces aparte de aforrar a construción do dispositivo. Reducirase así o número de compoñentes que o usuario ten que levar, xa que é habitual dispoñer dun móbil en todo momento.

Desenvolverase a aplicación para o sistema operativo Android xa que este é o sistema operativo máis empregado globalmente e nos permitirá chegar a un maior numero de usuarios.

Para suxeitar un teléfono móbil ao guiador da bicicleta existen múltiples ancoraxes que se adaptan a varios tamaños de teléfono. Tamén existen modelos configurables segundo o tamaño do teléfono que despois se poden imprimir en 3D.

Na figura 3.2 mostrase o posible aspecto da aplicación.

3.3 Esquema xeral de BikeCam

As esixencias principais deste dispositivo son a presenza de conexións para as luces LEDs, conexións para unha cámara e capacidade de procesamento e comunicación. Engadirase unha quinta esixencia, xa que o obxectivo do proxecto é conseguir unha solución de hardware e

Figura 3.2: Esquema do aspecto da aplicación BikeView.

software libre que posibilite que o usuario final poida adquirir os compoñentes e montalos de forma sinxela,polo que será preciso que os compoñentes sexan fáciles de adquirir e traballar con eles.

A continuación analizaranse as diferentes opción de hardware dispoñibles, as súas características e seus pros e contras para o proxecto proposto.

3.3.1 Placas de desenvolvemento

Sendo o desenvolvemento dunha placa dedicada a este propósito a solución idílica, o custo de este proceso xunto coa dificultade final da construción, sempre que non se optase por unha produción en serie, son as principais contras desta opción. Poren optarase por elixir unha placa cos requisitos requiridos entre as posibles opcións dispoñibles no mercado. Centrarémonos nos modelos co menor tamaño posible.

As placas contempladas son as seguintes:

- Arduino

  Arduino é unha compañía dedicada o deseño e produción de placas de desenvolvemento con software e hardware, o que permite que terceiras compañías produzan as súas placas permitindo prezos finais de produto moi baixos.

  Con un ide propio cunha linguaxe de programación baseada en C++, cunha ampla compatibilidade con diverso hardware, a diversidade de opcións e a sua facilidade de uso converte as súas placas nas máis populares do mercado.

  Polo seu tamaño e forma as placas Arduino consideradas son as seguintes:

- Arduino Micro e Arduino Nano

  O primeiro esta baseado no microcontrolador de 8bits ATmega32U4 cunha frecuencia de 16MHz, 32KB de memoria flash e 2.5KB de *SRAM*. Conta con 20 pins de entradas/saídas dixitais con 7 canles *PWM* e 12 pins de entradas analóxicas.

  O segundo baséase no microcontrolador de 8bits ATmega328 traballando a unha frecuencia de 16MHz, conta con 32KB de memoria flash e 2KB de *SRAM*. Conta con 22 pins de entradas/saídas dixitais dos cales 6 son *PWM* tamén conta con 8 pins de entradas analóxicas.

  Estas dúas versión o contar con gran cantidade de pins tanto dixitais como analóxicos e un consumo enerxético e moi baixo xunto cun baixo prezo inferior os 5 euros nas versións de fabricantes de terceiros fainos perfectos para o propósito de control de LEDs, pero a sua baixa potencia computacional dificultaría o procesamento de vídeo. Tampouco conta co hardware necesario para as comunicación sen fíos.

- Arduino MKR ZERO e Arduino MKR1000

  Ambos baseado no procesador ARM M0+ de 32 bit de baixo consumo cunha frecuencia de funcionamento de 48MHz, 256KB de memoria flash e 32kB de *SRAM*, contan con 7 entradas analóxicas e 1 saída analóxica e 12 pins poden funcionar como PWM. Inclúe conexións *SPI UART* e *I2C*. Tamén inclúen unha conexión para alimentalos directamente cunha batería de 3.7v. A diferencia entre ambos e que o primeiro conta con 22 pins de entrada e saída dixital mentres que o segundo conta con 8 pero inclúe un chip Wi-Fi.

  Ambos teñen as capacidades de procesamento necesarias para a xestión das luces, do vídeo e as conexións. O único punto negativo é que as cámaras compatibles a nivel de conexión e librerías con estas placas non dispoñen de moita calidade de vídeo.

  Estas placas poden obterse por entre 20 e 50 euros

- Arduino MKR VIDOR 4000

  Esta placa de desenvolvemento a parte do procesador ARM M0+ inclúe un chip *FPGA* que permite a sua configuración como diferentes hardware permitido que a placa poda dispoñer de diferentes compoñentes configurables como podería ser múltiples USB ou chips aceleradores de vídeo. Aparte conta con conexión micro HDMI mini PCI Express e un conector de cámara *MIPI* no que se poderían conectar diversas cámara con calidade máis que suficiente para este proxecto.

  O seu prezo e superior os 60 euros.

- Raspberry Pi

As Raspberry Pi son unha serie de placas de desenvolvemento cun prezo moi axustado e unha potencia suficiente para para pode executar un sistema operativo completo. Grazas a sua popularidade dispón dun amplo soporte e compatibilidade con diversos software e outras plataformas hardware que a fan perfecta para diversos proxectos, como robótica, *IoT* ou centros multimedia. A versión dispoñible con menor tamaño e a Raspberry Pi Zero

  - Raspberry Pi Zero e Raspberry PI Zero W

    Esta placa conta cun microprocesador baseado na arquitectura ARM de 32bits que funciona a 1GHz, acompañase de un procesador de vídeo e unha memoria ram de 512MB. No apartado de conexións conta con un micro USB de carga e outro de datos, unha saída de vídeo HDMI e outra analóxica, unha rañura para unha tarxeta micro sd e un conector de cámara CSI. Tamen conta con 20 pins de conexión que a dotan de entradas e saídas dixitais, dúas canles *PWM*, conexiós *SPI I2C* e *UART* xunto a conexións de 5v, 3.3v e terra. A versión Zero W tamén dispón dun chip Wi-Fi e Bluetooth. A sua potencia e capacidade de conexión a fan máis que capaz de para este proxecto, e o seu prezo, 5 e 10 euros respectivamente, é unha das súas principais vantaxes.

- ESP8266 e ESP32

  A principal característica destas placas é que implementan chips Wi-Fi e Wi-Fi máis Bluetooth respectivamente,contan cun procesador *RISC* de un ou dous núcleos con velocidades dispoñibles entre os 80MHz e 240MHz e memorias ram de entre 32KiB e 520KiB.

  Os seus múltiples portos e interfaces,*SPI* , *I2C*, *UART*, *PWM* entre outros, o seu baixo consumo e a sua compatibilidade co entorno de programación de arduino fainos ideais para pequenos proxectos de *IoT*, robótica ou domótica. Segundo as súas características poden obterse dende o prezo de un euro.

  O igual que pasaba coas placas Arduino os ESP son ideais para a parte do manexo das luces pero non para a xestión do vídeo. Estudarase como opción para a implementación do dispositivo BikeLed.

- Outras placas baseadas en procesadores ARM

  NO mercado existen múltiples placas de desenvolvemento baseadas en procesadores ARM, non obstante o prezo e o soporte da Raspberry Pi faina a mellor opción para a maioría de proxectos máis xenéricos.

- Outras placas baseadas en *FPGA*

Os chip *FPGA* permiten un nivel de personalización hardware moi elevado, pero tamén contan cun alto prezo, e na maioría dos casos cun *toolchain* privativo que e necesario pagar para poder desenvolver en eles. O contrapunto a sua versatilidade é un maior custo de desenvolvemento en comparación con solucións de programación de alto nivel.

Tendo en conta o tamaño, a potencia, as conexións dispoñibles e o prezo, decidiuse optar pola Raspberry Pi Zero e Zero W como a placa encargada do control das luces e do vídeo. O dispor esta dunhas características estándar poderíase substituír nun futuro por outro tipo de placa garantido así a modularidade do sistema.

3.3.2 Luces LED

As dúas principais vantaxes das luces LED fronte a outras formas de iluminación son o seu baixo consumo ese o seu pequeno tamaño. Estas calidades fainas ideais para un dispositivo portátil coma o que pretendemos construír. Os requirimentos principais son poder controlar a intensidade dos LEDs e dispoñer de polo menos un color vermello para indicar a posición e a freada, e un color amarelo ou ámbar para os intermitentes.

Coa intención de miniaturizar optarase por utilizar LEDs *RGB* que permiten xerar diversas combinacións de cores e así poder utilizar os mesmos LEDs para as diferentes funcións. Xa que a Raspberry Pi non conta con saídas analóxicas, utilizaranse as canles *PWM* que permite enviar sinais moduladas en pulsos. A modulación *PWM* permite acender e apagar os LEDs múltiples veces a unha alta frecuencia a unha velocidade, tan rápidas que o ollo humano percibe como diferentes intensidades lumínicas en función da anchura dos pulsos. En LEDs *RGB* compatibles o *PWM* tamén se pode utilizar para codificar a cor elixida, e en series de LEDs conectados e direccionables se pode elixir que LED a iluminar e a súa cor e intensidade individualmente, permitindo así controlar un alto numero de LEDs cunha soa saída *PWM*.

Existen diferentes tipos de LEDs *RGB* direccionables no mercado, elixiremos o tipo de LED en función do seu tipo de conexión, a dispoñibilidade de librerías de software compatibles e coa limitación de que deberán operar a 5v que a voltaxe constante que necesita a Raspberry Pi para funcionar.

Os tipos de LED direccionables analizados son:

- WS2812B e WS2813

  Estes LEDs inclúen un circuíto integrado en cada LED que o conectalos en serie permite o control dunha secuencia teoricamente infinita de LEDs. Cada LED conta con tres entradas e tres saídas: voltaxe, terra e datos. A información a pasar os LEDs se formara cun fluxo de datos a almacenar nun *buffer* en memoria, ocupando a información de cada un 3 bytes, e se pasarán o primeiro LED que lerá os primeiros 24 bits coa información

da intensidade de cada cor, vermella, verde e azul, e pasará o resto de datos o seguinte LED.

Cada LED tarda 30 microsegundos en recibir os datos e 50 microsegundos e actualizar a sua cor, o atraso de transmisión entre LEDs e de 0.5 microsegundos. O consumo máximo de cada LED e de 60 mA a 5V.

Os WS2813 engaden unha segunda liña de datos para que se un LED deixa de funcionar os seguintes poidan seguir recibindo a información.

- SK6812

  Estes LEDs comparten a maioría de características dos WS2812B coa diferencia de que aumenta a sua taxa de refresco a 1.2KHz con respecto os 400Hz dos WS2812B. Tamén engaden unha cuarta cor branca en cada LED.

  O a nivel de software son compatibles con WS2812B pero as súas diferenzas non permiten a sua interconexión física.

- APA102 e APA102C

  Estes LEDs contan cunha interface SPI que conta cunha sinal de datos é outra de reloxo. Isto e para solucionar o problema de sincronización que se poden producir cando os LED son manexados dende placas con capacidade de multitarefa sen un *kernel* especifico para entrada e saída a tempo real. Tamén aumenta sua taxa de refresco ata os 19.2kHz.

Sendo os LEDs APA102 superiores en características os outros dous, contan coa desvantaxe de que requiren máis cables de conexión. As súas vantaxes a nivel de velocidade e sincronismo son esencias para a xeración de imaxes ou vídeo pero non para simples animacións como as que utilizaremos neste proxecto. Porén optaremos por utilizar os LEDs WS2812B e WS2813 ou SK6812.

Este tipo de LEDs están dispoñibles en diferentes combinacións: LEDs individuais, tiras flexibles, tiras ríxidas, aneis e matrices. Faremos probas con tiras e aneis de diferentes tamaños.

3.3.3 Cámara

No caso da cámara plantexanse dúas opcións utilizar unha cámara usb ou unha das cámaras deseñadas para funcionar coa Raspberry Pi que utilizan a sua conexión *CSI*. Optaremos pola segunda opción xa que unha cámara usb implica un tamaño demasiado grande para o proxecto e ademais non soen estar indicadas para a iluminación de exteriores.

No mercado existen diversos módulos de cámara para a Raspberry Pi pero a maioría están baseados nos Raspberry Pi Camera Module V1 e Raspberry Pi Camera Module V2 a principal

diferencia entre ambos e que o primeiro conto con 5 megapixeles mentres o segundo conta con 8 megapixeles e unha notable mellora na calidade de imaxe.

As principais diferenzas nos módulos dispoñibles no mercado son:

- Tamaño

  Existen versións especificas para a Raspberry Pi Zero máis pequenas pero solo do Camera Module V1, as versións normais xa contan cun tamaño moi axustado.

- Presenza de filtro infravermello

  As cámara soen contar cun filtro de luz infravermella para evitar o *aliassing* que se produce nas cámara xa que as pantallas que utilizamos non están destinadas para emitir infravermellos e os nosos ollos non son capaces de percibilos. As cámaras que non contan con este filtro dan como resultado imaxes máis luminosas e cunha tonalidade violácea. Estas cámara son útiles para entornas exteriores no solpor ou para visión nocturna se se conta con fontes de luz infravermellas.

- Tipo de lente

  A uso dunha cámara cunha lente curva permite ampliar o campo de visión da cámara, se a lente e demasiado curva se producirá unha distorsión da imaxe nos bordes.

  Xa que estas cámaras son todas compatíbeis a nivel de software probaremos diversos tipos con varios tipos de lente xa sexan incorporados ou engadindo unha lente externa.

3.3.4 Alimentación e baterías

A Raspberry Pi Zero necesita unha fonte de alimentación que provea de 5V constantes. O seu consumo enerxético varía segundo a carga computacional, o uso do Wi-Fi e o uso da cámara podendo ascender a entorno 300mA. Os LEDs tamén funcionarán a 5V cun consumo máximo de 60mA por LED, cando emiten luz branca a máxima intensidade.

Plantexamos dúas solución posibles:

- Bateria externa USB

  Utilizar unha batería usb externa permite dispoñer de altas capacidades que prolongarían o tempo de uso pero implican o uso dun dispositivo a maiores. Outra de desvantaxes e que cando se esgote a batería a corrente interrompese de golpe sen que a Raspberry poida realizar un apagado normal, como consecuencia tras varios apagados podería danarse o sistema de ficheiros se se estaba a escribir nel no momento do apagado.

- Batería, circuíto de alimentación e circuíto de acendido e apagado.

  A maioría de baterías usadas en electrónica son baterías de *ions de litio* principalmente devido as sua alta capacidade enerxética e lonxevidade. A *voltaxe nominal* destas baterías e de 3.7V sendo 4.2V a voltaxe coa carga o máximo e por debaixo de 3V deixan de proporcionar suficiente intensidade eléctrica para a maioría de aplicacións. Este tipo de baterías son as que atoparemos nos teléfonos móbiles, ordenadores portátiles e incluso en vehículos eléctricos. Poden colocarse en serie cando e necesario unha maior voltaxe ou en paralelo cando o que se necesita e unha maior capacidade.

  Para poder utilizar estas baterías é necesario un circuíto de carga, un de protección, e un de conversión de voltaxe. Na maioría dos casos as batería de consumo utilizan o estándar USB, que funciona a 5V, tanto para cargarse como para proporcionar enerxía. Polo que será necesario un chip de carga que acepte unha toma de 5V e que cargue a batería ata 4.2V. As baterías de litio poden ser perigosas danándose e chegando incluso a estoupar se se descargan demasiado ou se se sobrecargan polo que é necesario un circuíto de protección que evite a sobrecarga e a sobrecarga. Para proporcionar unha saída estable de 5V tamén é necesario un conversor de voltaxe. É habitual atopar o circuítos de carga máis protección xuntos no mesmo chip aínda que tamén se atopan circuítos que integran as tres funcionalidades.

3.3.5 Software a utilizar

A Raspberry Pi conta cunha ampla gama de sistemas operativos, algúns con propósitos concretos como reprodución de multimedia, servidores locais ou nodos de rede. Tamén conta con versións das distribucións Linux máis popular como Ubuntu, Arch ou Kali entre outros. Raspbian é unha distribución baseada en Debian, é máis antiga e máis optimizada para a Raspberry Pi e a que dispón de máis soporte polo que será a elixida para o proxecto.

Para o control dos LEDs existen varias librarías dispoñibles para varias linguaxes de programación. As máis utilizadas son a de Adafruit, que só está dispoñible en Python, e a de Jeremy Garff que será a que utilicemeos, xa que conta con unha documentación detallada e esta dispoñible en varias linguaxes de programación, entre outras C, Python e Java.

Para a captura e transmisión do vídeo contamos con varias alternativas. A libraría *picamera* para Python permite configurar calquera parámetro e o *streaming* do vídeo na rede. Por outra parte o software para captura de vídeo *raspivideo*, escrito en Python, tamén permite moitos parámetros de configuración é a sua saída de vídeo pode enviarse a rede utilizando algún programa para redirecionar o *bitstream* dos datos como pode ser o software *socat*.

A comunicación entre o dispositivo de control e o de luces e captura de vídeo realizarase a través dunha conexión IP. Para manexar as peticións podemos empregar unha das clases servidor Python, unha libraría de terceiros ou implementar o servidor a nivel de *sockets*.

Por motivos de compatibilidade entre todo o software necesitado para o control de LEDs, vídeo, e conexións, decidiremos integralo todo nunha aplicación Python que se encargará das tres tarefas.

3.3.6 Caixa e ancoraxes á bicicleta

O lugar a colocar o dispositivo de iluminación e captura será na barra da sela da bicicleta, esta posición é a ideal tanto para capturar o vídeo como para que as luces sexan vistas polo tráfico que circula detrás do vehículo.

A Raspberry Pi conta cunha caixa oficial na que se pode instalar xunto coa cámara V2. Esta é a que utilizaremos no caso de alimentala cunha batería externa. Para suxeitar a placa á barra deseñaremos un soporte e o imprimiremos cunha impresora 3D. Para a versión con batería interna deseñaremos unha caixa protectora que albergue tódolos compoñentes e que se poida suxeitar á barra.

Os prototipos imprimiranse en *PLA* un material biodegradable e de fácil impresión, no é o mellor material para resistir a auga ou a humidade pero é ideal para o prototipado. Poderá utilizarse *ABS* para imprimir unha versión final, un material moito máis resistente as condicións atmosféricas.

3.3.7 Esquema conceptual de BikeCam

As decisións tomadas condúcenos a un dispositivo composto por unha Raspberry Pi Zero con presenza de batería interna ou externa e unha o varias tiras LED RGB, na figura 3.3 mostrase unha posible aproximación incluíndo unha Raspberry Pi Zero W unha Pi Camera V1 e unha tira e un anel WS2812 de 8 luces cada un.

3.3.8 Esquema conceptual de BikeLed

Este dispositvio cotaría con un Arduino Nano mais un chip Wi-Fi ou Bluetooth ou cun ESP32 conectado a unhas tiras Leds RGB e unhas Batería, na foto da figura 3.4 mostras unha imaxe dos posibles compoñenetes que poderia implemtear, con un anel RGB de 8 luces, unha bateria de 400mA e un microcontrolador ESP8266 Wemmos D1 mini con Wi-Fi.

3.4 Comunicacións entre BikeCam e BikeView

Optarase por utilizar un sistema de conexión cliente servidor no que BikeView realizará peticións o servidor en BikeCam que responderá actuando as luces e devolvendo cofirmacións e a sinal de vídeo.

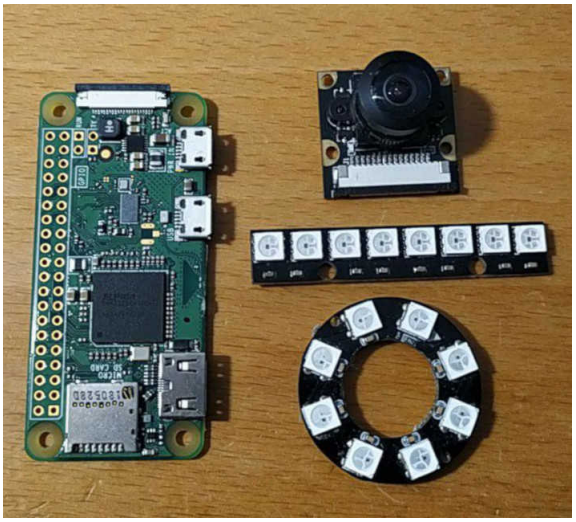

Figura 3.3: Exemplo de implementación de BikeCam.

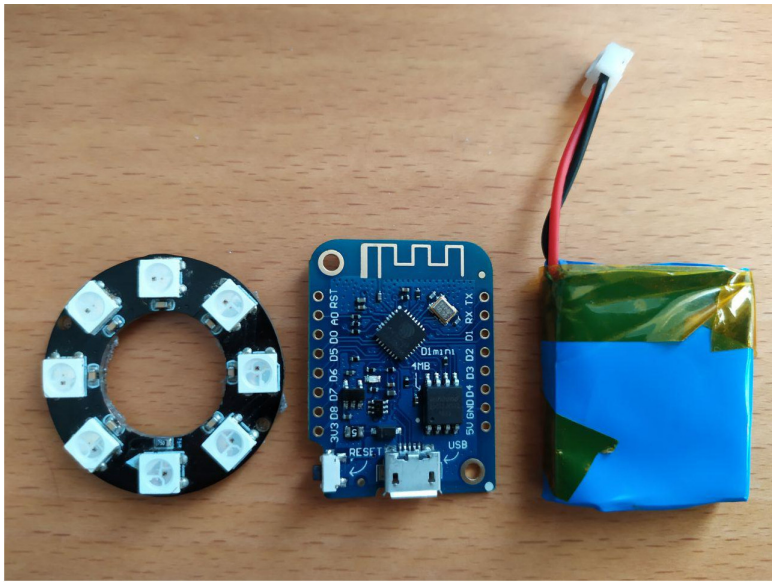

Figura 3.4: Exemplo de implementación de BikeLed.

3.4.1 Canle de comunicaións

Partindo da elección da Raspebrry Pi para BikeCam e un dispositivo Android para BikeView as opcións para a implementación física da conexión son as seguintes. begin

- USB

  A Raspberry Pi Zero conta cun conector USB 2.0, e os dispositivos Android implementan este estándar ou o superior 3.0 polo que a velocidade máxima será restrinxida pola Raspberry Pi permitindo un máximo de 480Mbits/s esta velocidade permitiría a transimisión de vidio sen problemas. Tanto no dispositivo Android como na Raspberry Pi pódese implementar unha conexión Ethernet virtual sobre USB o que facilitaría a estandarización das comunicacións.

  Unha das principais vantaxes de usar unha conexión USB é que poderíase alimentar enerxeticamente o dispositivo BikeCam dende BikeView sempre e cando o consumo de este non supere o máximo de 500mA que pode subministrar un dispositivo Android por USB en modo OTG.

  A principal desvantaxe é a necesidade de ter un cable unindo ambos dispositivos dende o guiador a sela da bicicleta.

- WI-Fi

  A Raspberry Pi Zero conta con un chip WI-Fi 802.11n e calquera móbil actual implementan esta versión do estándar ou unha superior. Este estandar Wi-Fi permite unha velocidade máxima de transmisión de 300Mbits/s tamén neste caso suficiente para a transmisión de vídeo HD.

  A principal vantaxe do uso do WI-Fi é a independencia dos dispositivos pero coma contrapartida será necesario que BikeCam conte cunha batería propia.

- Bluetooth

  Bluetooth 4.0 é a versión do estándar implemetado pola Rasberry Pi Zero W e neste caso tamén marcará a velocidade máxima de transmisión xa que os dispositivos Android actuais implementan versións iguales ou superiores do estándar.Esta velocidade será 25Mbits/s o que non nos permitiría transmitir vídeo en tempo real. Sen embargo esta conexión e máis que suficiente para o dispositivo BikeLed que non require transmisión de vídeo.

  Unha das súas vantaxes fronte o Wi-Fi e menor consumo enerxético.

Tendo en conta a viabilidade das tres opción optarase por a implementación das conexión a través de Wi-Fi xa que é a opción que menos limitacións carrexa. Neste caso optarase por

usar modo *hotspot* de Android no que o dispositvo BikeView xerará unha rede Wi-Fi a que se conectará io dispositivo BikeCam.

3.4.2 Protocolos de comunicación

Establecemento da conexión

Para permitir que BikeView se conecte a BikeCam sen ter que coñecer a dirección *ip* de este procederase da seguinte maneira. Dende o servidor crearase un *thread* no que se abrira un *socket* encargado de enviar unha mensaxe a dirección de *broadcast* para que poda ser recibido por tódolos dispositivos da rede. Por outro BikeView abre un porto a espera de captura este paquete de *broadcast* unha vez obtido collerase a dirección IP do remitente e crearase un conexión con BikeCam a través dun *socket UDP*.

Transmisión de ordes

Como o esquema de comunicacións involucra so a dous dispositivos intentarase realizar un protocolo de comunicación o máis sinxelo posible. Será o dispositvo BikeLed o que exerza de servidor e escoite as ordes enviadas dende BikeView, para elo a primeira aproximación foi a creación dun servidor *http* pero como non requiríamos da necesidade de utilizar múltiples threads decidiuse facer unha aproximación máis simple, implementar un protocolo de envío de ordes a través de paquetes *UDP*.

O dispositivo BikeView enviará ordes de control e ordes para saber se o sistema segue activo, e o servidor de BikeCam se recibiu a orde responderá devolvendo a mesma mensaxe. Este é o mecanismo que utilizará BikeView para saber se o servidor segue en funcionamento.

Cando BikeCam recibe unha mensaxe comproba se correspondese con algunha das ordes preestablecidas, se é asi fará unha chamada a función solicitada, se a mensaxe non se corresponde o servidor interpretará que se ha requirido o peche da conexión e

As mensaxes serán as seguintes:

- **r** para o xiro á dereita.
- **l** para o xiro á esquerda.
- **n** para a luz de noite.
- **b** para o padrón de freo.
- **k** para o padrón de intermitencia.
- **o** para apagar as luces.
- **v** para iniciar a captura e transmisión de vídeo.

- **w** para deter a captura e a transmisión de vídeo.
- **c** para comprobar que a conexión segue aberta.
- **valor numérico** para establecer a intensidade das luces.

3.4.3 Protocolos de videostreaming

Existen varias opcións para transmitir vídeo a trabes da rede pero moi poucas están orientadas a transmisión de vídeo en tempo real e as que o conseguen e a base de reducir a calidade do vídeo.

As opcións que maior velocidade prometen son aquelas que transmiten o vídeo capturado en formato H.264 [10], o estandar de codificación de vídeo utilizado pola maioría de formatos de vídeo, directamente e redireccionao a rede en paquetes *UDP*. No capítulo 4 detallaranse a probas realizas a diferentes software ata obter a menor latencia de transmisión.

Para a recepción realizada no dispositivo BikeView levaremos a cabo a decodificación sen utilizar ningunha librería descompoñendo os paquetes recibidos, descodificándoos cun *MediaCodec* e enviando os fotogramas a superficie de vídeo.

Capítulo 4

Deseño e implementación de BikeCam

Neste capítulo afondarase no desenvolvemento do dispositivo comezando polo aspecto físico profundando na construción do dispositivo e a implementación do sofware tendo en conta os problemas xurdidos e as solucións aplicadas.

4.1 Estrutura xeral de BikeCam

O dispositivo BikeCam componse de catro módulos principais. Son os seguintes.

- Rasberry Pi Zero

  É o módulo principal sobre o que se apoiarán o resto. A súas funcións principais son as do procesamento, as comunicacións e o control do resto de módulos. Depende dunha alimentación de 5V para funcionar.

- LEDs

  Este módulo componse como mínimo dunha ou vairas series LEDs RGB direcionables. Para funcionar necesita alimentación a 5V e unha liña de datos conecta a Rasberry Pi, pode requirir un conversor lóxico de nivel nesta conexión.

- Cámara

  Este módulo se conecta directamente a Rasberry Pi, da que recibe alimentación e control e devolve o vídeo capturado.

- Alimentación

  Este módulo pode implementarse de duas maneiras.

  - Unha batería USB que alimenta a Raspberry Pi, e esta á súa vez os LEDs e á cámara.

– O módulo LipoPi, encargado da alimentación enerxética do dispositivo na versión con batería interna. Conta cun chip de alimentación Adafruit Powerboost 1000C un pulsador e unha serie de resistencias diodos e condensadores xunto cunha batería de 3.7V. Proporciona alimentación á Rasberry Pi e ós LEDs, utiliza unha conexión coa Rasperry para informar do estado da batería, outro pin para informar a Raspberry do premido do pulsador e unha ultima liña de conexión coa que a Raspberry mantén o chip aceso.

4.2 LEDs

Crearanse dous prototipos con dúas configuracións de LEDs diferentes, ambas contarán cun anel de 8 LEDs *RGB* direccionables WS2812B, que conta cun tamaño perfecto para colocar arredor da cámara, a segunda opción contara ademais con dúas tiras de 8 LEDs cada unha que se utilizarán como indicadores de xiro para aumentar a visibilidade.

4.2.1 Conexión coa Raspberry Pi

Estes LEDs contan con catro puntos de conexión entrada de voltaxe, terra, entrada de datos é saída de datos.

A voltaxe necesaria para alimentalos é de 5V, aínda que na maioría dos caso o fabricante indica un soporte a voltaxes de entrada de entre 4V e 7V. A Raspberry Pi conta con pins de saída a 5V conectada directamente á entrada, sen contar coa limitación dun fusible como noutras versións da placa, polo que pode alimentar os LEDs directamente pero xa que cada LED pode chegar a consumir ata 60mA e os pins de 5V da Raspberry poden proporcionar ata 300mA [11] e facer pasar polo *power rail* unha corrente excesiva podería provocar danos ou unha aumento da temperatura da placa implicando menor velocidade e maior consumo. Tendo en conta que o fabricante non recomenda que a placa consuma máis de 1A será conveniente alimentar os LEDs directamente dende a fonte de alimentación, especialmente na versión na que utilizaremos 24 LEDs.

Para a conexión de datos terase que usar unha das saídas da placa conectadas a un das dúas canles *PWM* da que dispón. Estas saídas lóxicas contan cunha voltaxe de 3.3V, pero as tiras LEDs requiren que a voltaxe na entrada de datos, para ser interpretada como un valor lóxico HIGH, sexa polo menos un 70% da voltaxe de alimentación, neste caso 3.5V. Para solucionalo proponse dúas opcións, reducir a voltaxe de entrada dos LEDs, o que implicaría unha menor luminosidade, ou aumentar a voltaxe do valor lóxico, optarase por esta solución utilizando un conversor lóxico de nivel. Na práctica comprobaremos que os LEDs utilizados seguen interpretando como valor lóxico positivo os 3.3V polo que a utilización ou non do conversor de nivel a valoraremos máis adiante en función do espacio dispoñible.

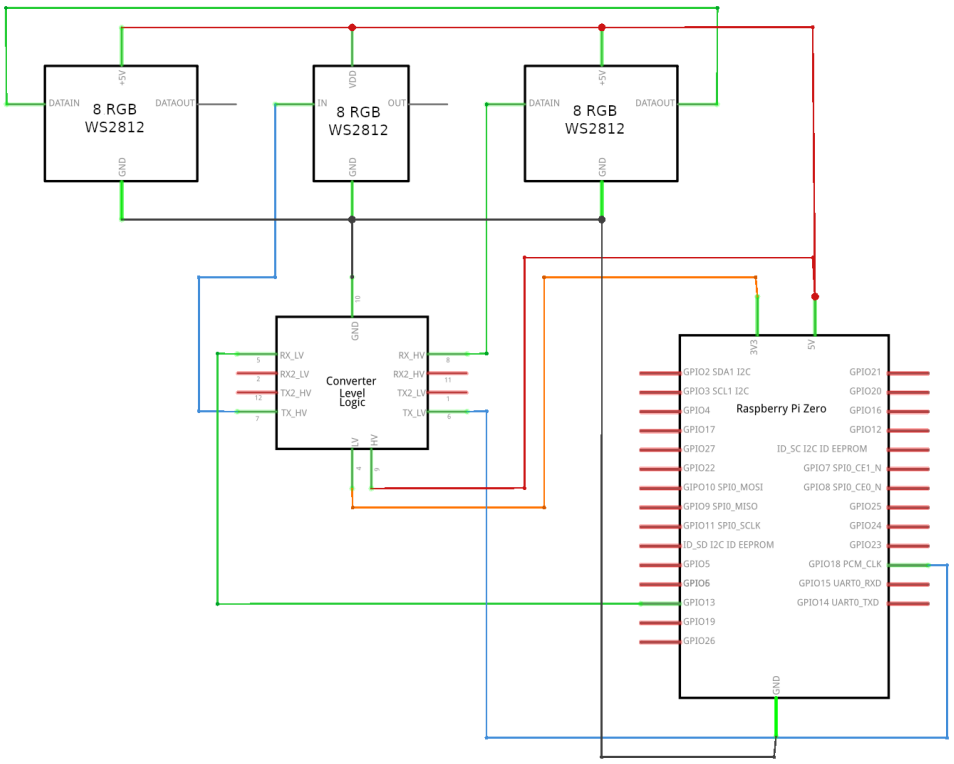

Figura 4.1: Diagrama de conexión dos LEDs.

Os pins da placa dispoñibles con conexión *PWM* serán o GPIO18 e GPIO12 para a canle PWM0 e GPIO13 para a canle PWM1. A libraría que utilizaremos [12] tamén permite controlar o LED mediante a conexión SPI e a PCM, utilizaremos a *PWM* por que é a única que nos permite controlar dúas tiras LED independentes simultaneamente, coa contraindicación de que ao utilizar o *PWM* a Raspberry non poderá xerar audio analóxico, algo que non necesitarase neste proxecto.

Utilizaremos o GPIO18 para controlar o anel LED e o GPIO13 no caso que utilicemos os intermitentes que irán conectados un ao outro como se indica na figura 4.1.

4.3 Cámara

A cámara a utilizar é a Raspberry Pi Camera, probarase a versión 1 e a versión 2, ambas conectarase a Rasberry Pi Zero co mesmo cable, o conector da placa é delicado polo que deberase conectar con coidado. Para habilitala executarase o comando *raspiconfig* na terminal e no apartado de interfaces activarase a opción cámara.

Na táboa 4.3 móstrase unha comparativa de ámbalas dúas cámaras e na foto da figura 4.2

Táboa 4.1: Táboa comparativa da Raspberry Pi Camera V1 e V2 [4]

	V1	V2
Sensor	5 Mpíxeles	5 Mpíxeles
Resolución foto	2592x1944	3280x2464
Vídeo maxi	1080p	1080p
Tamaño do módulo	20x25x10 mm	25x23x9 mm

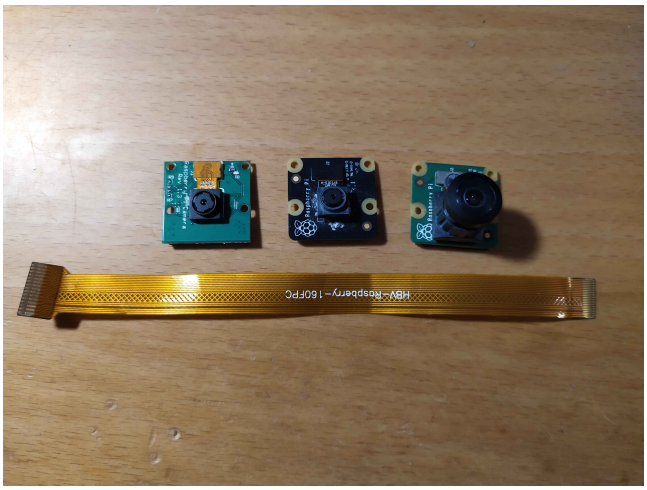

Figura 4.2: De esquerda a dereita a Raspberry Pi Camera Module V1, V2 sen filtro infravermello e V2 cunha lente de gran angular

poden verse as tres cámaras a utilizar, a versión 1.3 a versión 2.1 e a 2.1 sen filtro infravermello.

4.3.1 Lentes

Para poder capturar o completo da estrada a cámara necesitará de algún tipo de lente que permita un maior campo de visión. Existen versións da cámara que xa inclúen unha lente, pero tamén poderemos atopar lentes externas coma as destinadas para os dispositivos móbiles, que contan co tamaño necesario para a cámara da Raspberry. Na figura 4.3 móstranse as lentes a probar e analizar as tres superiores colocaranse sobre a cámara e as tres inferiores so para a versión da cámara que incorpora lente.

4.4 Alimentación e enerxía

O consumo de amperios da Raspberry Pi Zero sen carga de traballo é duns 120 mA, gravando vídeo a 1080p o consumo é de 230mA, gravarase vídeo a 720p polo que o consumo será

Figura 4.3: Lentes con diferente ángulo de visión. De esquerda a dereita: fila superior 0.67x, 180º, 0.4x, fila inferior 130º, 160º, 220º

algo menor pero engadirase o consumo do chip Wi-Fi funcionando. Os LEDs ws2812 teñen un consumo máximo de 60 mA cada un 20 mA como máximo por cada un dos tres LEDs *RGB* a máxima intensidade, o noso máximo consumo realizarse coa luz vermella acesa de forma continua, xa que no resto de modos os padróns de intermitencia reducen o consumo. Contamos con 24 destes LEDs polo que o consumo máximo será de 20mA por 24 LEDs, un total de 480mA que sumados o consumo da Raspberry Pi nos da un consumo máximo teórico de 710mA na versión sen LEDs intermitentes o consumo seria de 160mA máis 230mA, en total 390mA.

- Unha primeira versión máis sinxela contará solo cunha batería USB para alimentar a Raspberry. Os requisitos desta batería serán a amperaxe e a capacidade.

  Partirase do valor do consumo máximo aproximado de 400mA para calcular o tempo de funcionamento. Con esta amperaxe ós 5V que funcionan a Raspberry e os LEDs a potencia utilizada sería de 2W. Neste suposto unha batería de 5000mAh cunha voltaxe nominal de 3.7V pode proporcionar 18.5Wh polo que duraría ata 9 horas e 15 minutos, no caso dunha batería de 1000mAh o tempo mínimo teórico de funcionamento sería de algo menos de dúas horas. Comprobaremos se estes supostos se cumpren facendo medicións do tempo de funcionamento.

- Realizaremos unha segunda versión máis avanzada que apagará o dispositivo cando a batería baixe de certo limite de voltaxe para evitar que o dispositivo se desconecte e contará tamén cun pulsador para poder acendela e apagala.

  Para isto utilizaremos o chip de carga Adafruit Powerboost 1000 que conta cunhas características moi interesantes a maiores da protección de sobrecarga conta cunha LED

e un pin que se activará cando a voltaxe da batería baixe dos 3.2v, unha voltaxe operacional de 5.2v para evitar perdidas de voltaxe en cables e conectores, un pin habilitador que permite conectar ou desconectar a batería, e proporciona 1 amperio de intensidade sen baixar a voltaxe dos 5v. Non conta con protección de sobrecarga polo que as batería que utilicemos deben incluír un circuíto de protección, este é o caso da maioría de baterías, de utilizar unha sin protección, como unha cela 18650, deberemos engadir o circuíto de protección ou asegurarnos de implementar o apagado por voltaxe baixa correctamente.

A realización o circuíto basearase no guía *lipopi* de Daniel Bull [13] que utiliza o Adafruit Powerboost, nas súas dúas versións a de 500 mA e 1 A, para programar o apagado automático da Rapberry Pi cando a batería baixe de 3.2v e un pulsador para o acendido, tamén conta con dúas versións máis unha que tamén permite o apagado, e outra que monitoriza a voltaxe da batería. Realizarase a versión con pulsador para acendido e apagado.

O funcionamento é o seguinte, ao premer o pulsador conéctase o positivo da batería co pin abilitador, acendendo o Adafruit Powerboost e por conseguinte acendendo a Raspberry Pi. Ao acender a Raspberry Pi un pin conectado o pin habilitador do Adafruit Powerboost acenderase pare seguir mantendo un valor positivo. Para evitar que o voltaxe no pin habilitador caia no tempo entre que pulsamos o pulsador e a Raspberry Pi arranca e encárgase de manter o valor positivo, situaremos un circuíto RC formado por un condensador cunha resistencia en paralelo entre o pulsador e o pin habilitador. O condensador cargarase cando o pulsador cerre o circuíto e descargarase a continuación mantendo a voltaxe o tempo suficiente para que a Raspberry arranque e acenda o pin. Utilizarase un condensador de 100μF xunto cunha resistencia 100kΩ que proporcionan un tempo suficiente de 10 segundos. O pin da Raspberry que utilizaremos para este propósito pode ser o 14 correspondente a conexión *uart*, que se acenderá coa Raspberry e se desconectará cando se apague, engadiremos unha resistencia de 10kΩ para protexer este pin. Tamén poderíase utilizar calquera outro pin de propósito xeral indicando no arquivo de configuración *config.txt* na partición *boot* da Raspberry, que o pin arranque cun valor positivo e cun valor negativo cando o dispositivo se apague, no caso de utilizar o pin *GPIO* 5 as ordes serían as seguintes: $gpio = 5 = op, dh$ para que o pin arranque con valor positivo, $dtoverlay = gpio - poweroff, gpiopin = 5, active_low =$ "y" para deixar o pin apagado cando se apaga a Raspberry.

Para o apagado utilizarase un segundo pin conectado ao pulsador, cando este se pulse, estando a Rasberry acesa, se conectará a voltaxe da batería, cando este valor positivo chegue o pin un script Python encargarase de apagar o dispositivo. Engadiranse un divisor de voltaxe para reducir a voltaxe da batería xa que cando está completamente

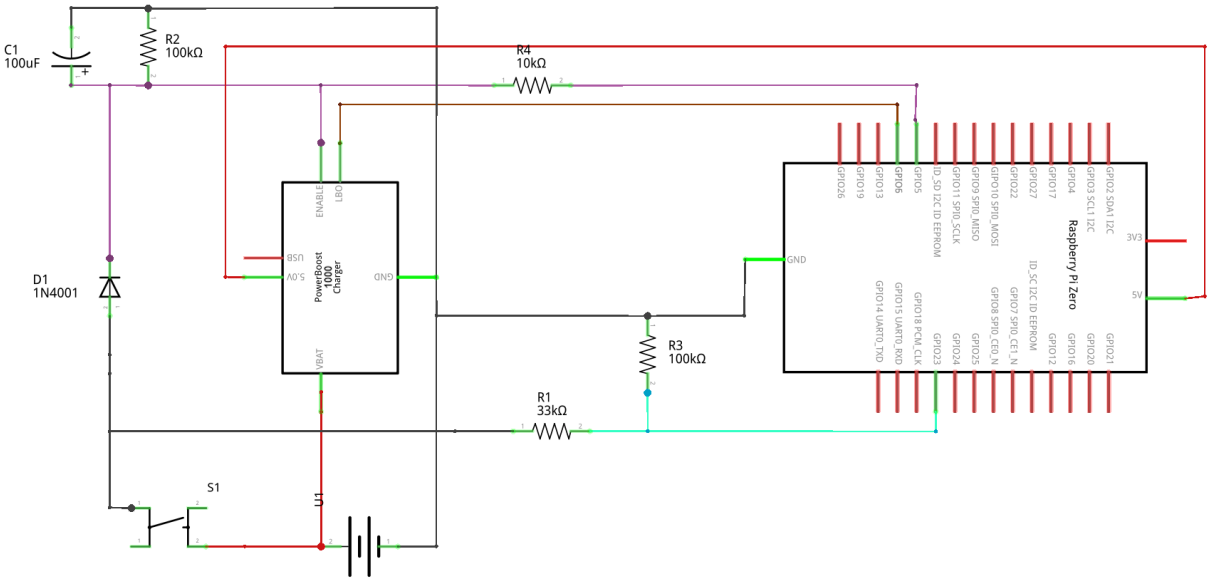

Figura 4.4: Diagrama de funcionamento do circuíto de alimentación.

cargada a súa voltaxe de 4.2v é superior ao máximo valor lóxico tolerado pola Raspberry Pi de 3.3V. Utilizaremos unha resistencia de 33kΩ conectada entre o pin e a batería e unha resistencia de 100kΩ entre o pin e terra. Para evitar que o pin de acendido dispare o apagado situarase un díodo entre o pin de acendido e apagado evitando que a voltaxe circule nesa dirección.

Finalmente conectarase o pin indicador de batería baixa a outro pin de entrada da Raspberry que mediante o script Python apagará o dispositivo. O diagrama de funcionamento mostrase na figura 4.4.

Na figura 4.5 podemos ver o diagrama final do dispositivo e na figura 4.6 o esquema completo integrando os LEDs o circuíto de carga e a cámara.

Na implementación física colocaremos o chip de carga xunto cos compoñentes electrónicos nunha placa cun conector para poder conectalo directamente os conectores da Raspberry Pi Zero. Tamén colocarase nesta placa o conversor lóxico de nivel como se ve na figura 4.7

Na elección da batería teremos en conta a maiores da súa capacidade o seu tamaño e forma, sendo o ideal que sexa similar ao da Raspberry para poder integrala no dispositivo con facilidade. Por eso elixiremos unha batería de 1600mA e 5.92Wh con circuíto de protección e unhas dimensións de 9 x 34 x 50mm que para este suposto, xa que engadindo os LEDs extra calcúlase un consumo máximo de 3.55W, debería proporcionar

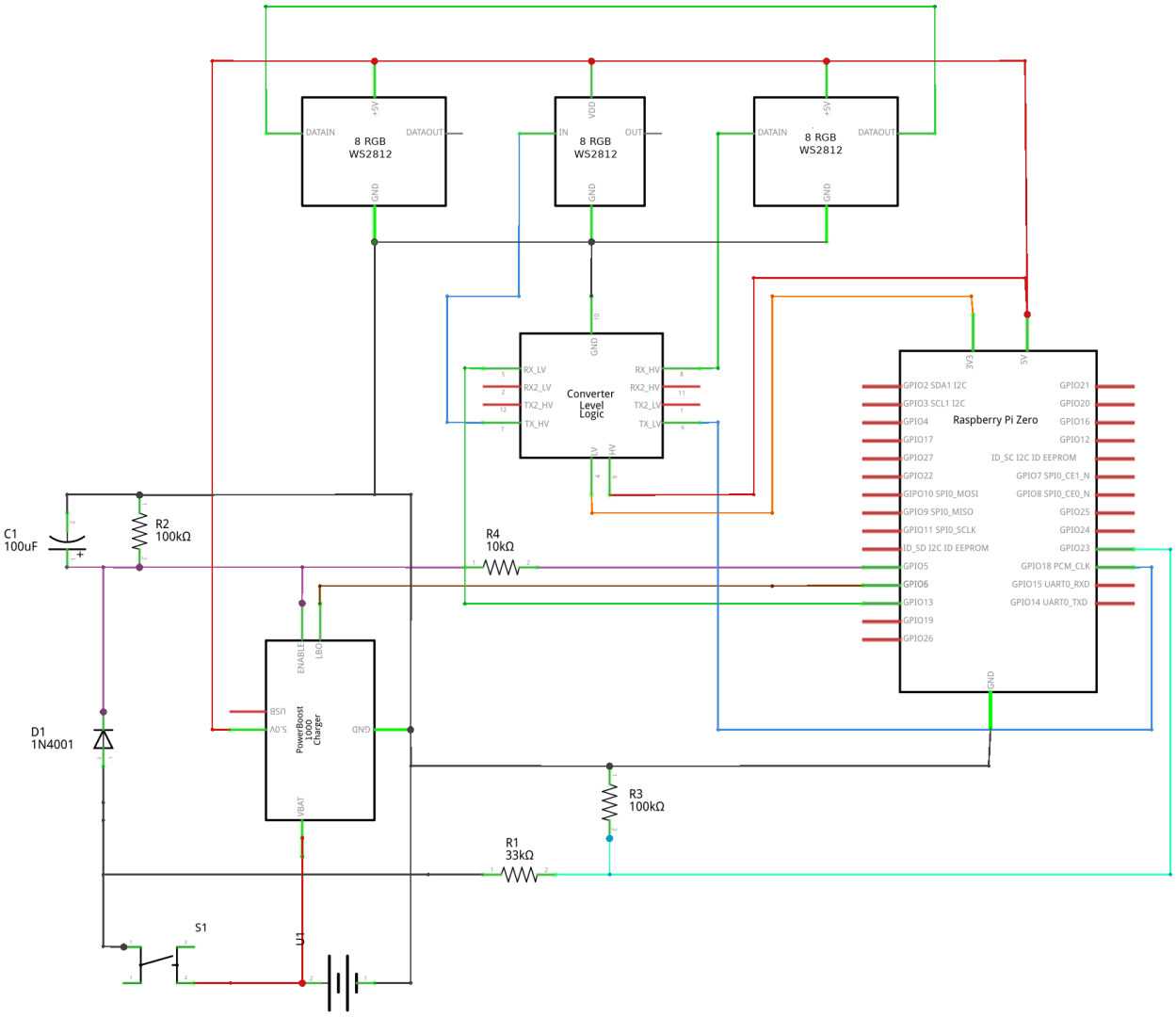

Figura 4.5: Diagrama do dispositivo.

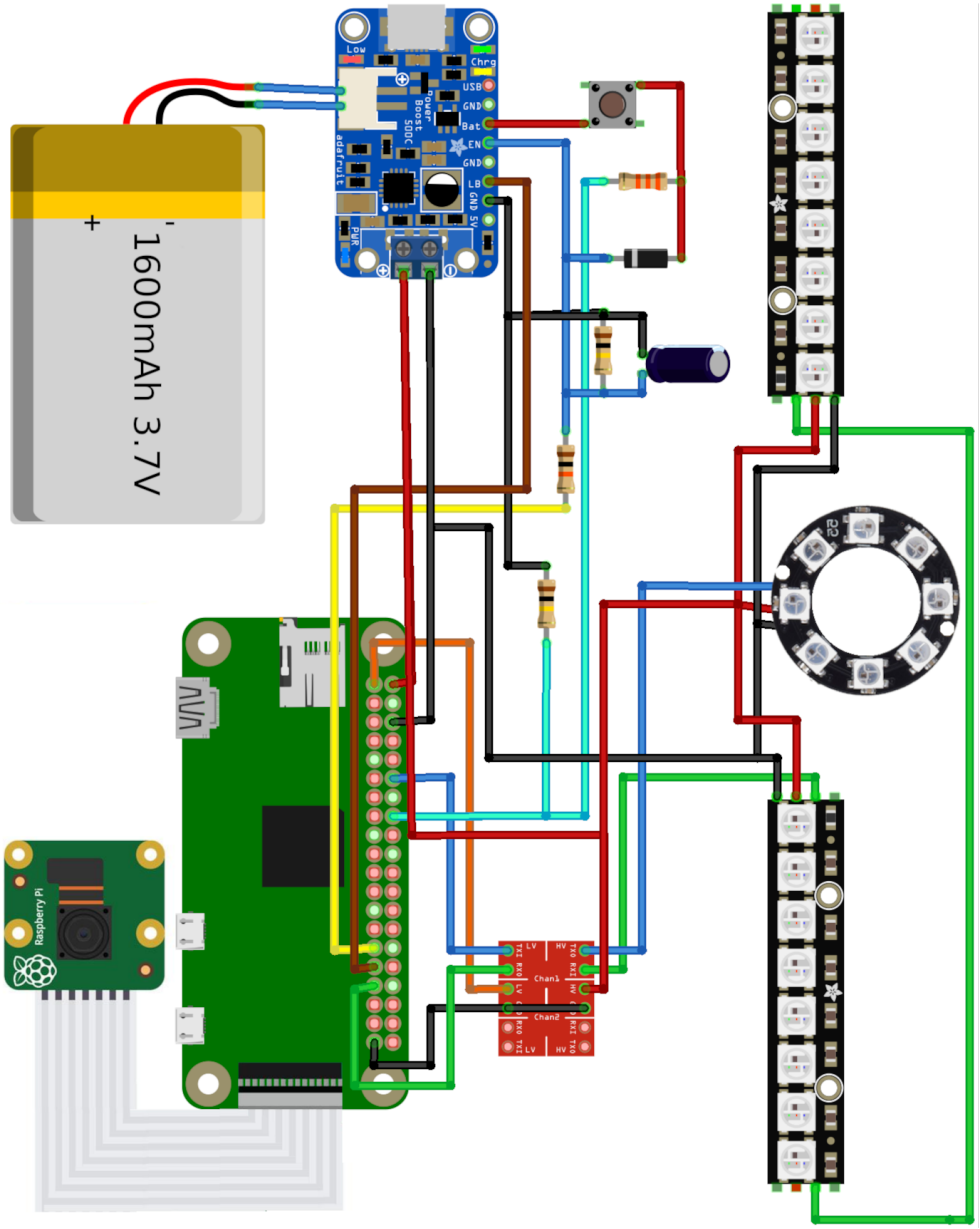

Figura 4.6: Esquema completo do dispositivo.

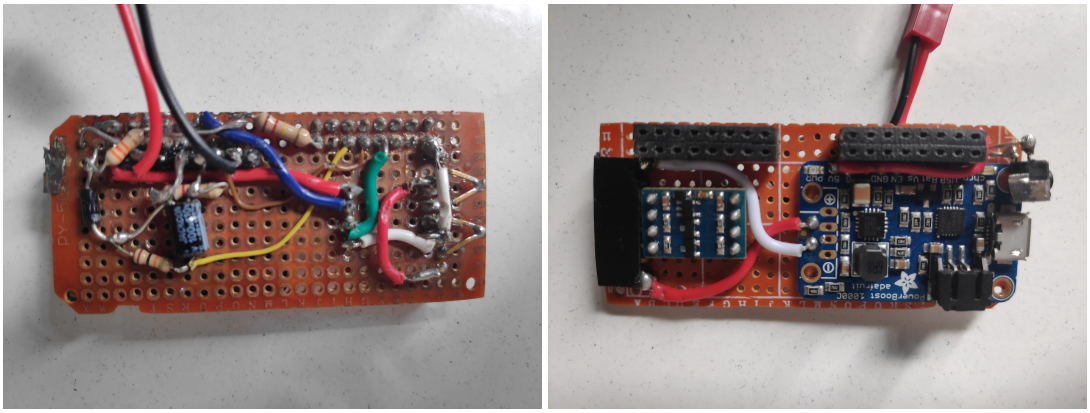

Figura 4.7: Imaxes do circuíto por ambas caras.

un tempo mínimo de funcionamento de 2 horas aproximadamente.

4.5 Software de control de BikeCam

Nesta sección abordarase todo o software utilizado en BikeCam. Este estruturase en torno ao programa Python *lightsServer* un servidor que se encargará de controlar as luces e executar a orde de transición e parada do video.

No caso de contar cun circuíto de alimentación LipoPi, tamén será necesario ter en funcionamento o script Python LipoPi correspondente.

4.5.1 lightsServer

Este é o servidor principal, as súas funcións son compartir a dirección IP de BikeCam na rede mediante *broadcasting*, inicializar e controlar os LEDs a través da libraría rpi_ws281x, aceptar conexións de BikeView e recibir, executar e confirmar as súas ordes.

Broadcasting da dirección IP

Os dispositivos conectaranse mediante unha rede local, xa sexa a través dun cable USB ou dunha rede Wi-Fi, en ámbolos dous casos o dispositivo Android será o encargado de aloxar a rede xa sexa compartindo por USB ou creando un punto de aceso Wi-Fi.

En canto arranque o servidor execútase a función encargada de *broadcastear* cada 3 segundos unha mensaxe co texto ”BikeView” á dirección de *broadcast* mediante un *datagram socket*. Esta función deterase se o servidor acepta unha conexón entrante. No momento que se peche a conexión, chamarase á función de novo e seguirase *broadcasteando* á espera dunha nova conexión.

Recepción de ordes

Para recibir os comandos enviados dende o dispositivo móbil e executalos mediante Python implementaremos un servidor tamén Python para poder integrar a recepción e a execución de ordes no mesmo programa.

A primeira opción será utilizar peticións *http* e manexalas mediante a clase de Python *BaseHTTPRequestHandler*, o problema é que esta clase bloquea o programa mentres se executan as ordes correspondentes á petición recibida. Para solucionar este problema poderiáse implementar un servidor *multithread* ou utilizar unha libraría para Python que implemente o servidor *multithread* de forma transparente. Plantexase utilizar as librarias *multithread Tornado*, *Twisted* ou *lighttpd* pero debido ós recursos limitados da Rasberry Pi Zero o uso dunha destas librarías podería implicar maiores latencias e consumo enerxético.

Finalmente optase por manexar a conexión directamente mediante *sockets* non bloqueastes é así poder utilizar un solo *thread*. Para elo introduciremos as peticións de conexión nunha lista, unha vez aceptada introducirase nunha segunda lista as mensaxes recibidas e se executara a orde correspondente para cada mensaxe.

Xestión dos LEDs

Para manexar os LEDs utilizaremos a libraría *rpi_ws281x* de Jeremy Garff [12] na súa versión para Python. O seu funcionamento é moi simple, primeiro teremos que configurar os parámetros da tira LED, coma o número de LEDs, o pin a que está conectado, o tipo de tira ou a canle *PWM* entre outros. No programa principal deberemos inicializar os LEDs con estes parámetros e executala coa función *begin*. Cada vez que queiramos que os les cambien os seu estado chamaremos a función *show*.

Escribiranse funcións encargadas dos padróns de iluminación. Estes padróns serán os seguintes:

- **Luz vermella fixa**, É a encargada de indicar a posición da bicicleta.
- **Luz vermella intermitente**, A sua función é a mima que a anterior, pero o padrón de intermitencia aumentará a visibilidade. Crearanse distintos padróns combinando distintas frecuencias en intensidades lumínicas.
- **Luz vermella incremental**, É a encargada de indicar a freada, a súa intensidade aumentará ata o valor máximo para emular as luces de freada dos coches.
- **Luz amarela de xiro á esquerda ou dereita**, Indica o xiro iluminando progresivamente os LEDs do anel dende os situados no centro ata os do estremo esquerdo ou dereito, de dispoñer das tiras extra de LEDs de xiro estas iluminaranse a continuación. Unha vez iluminados tódolos LEDs estes se apagarán e o padrón repetirase de novo.

Tamén se escribirá unha función para controlar a intensidade dos LEDs en función dun valor numérico recibido, 0 será mínimo e 9 a máxima intensidade.

4.5.2 Captura de vídeo

Raspivideo é o software que utilizaremos para capturar as imaxes, o programa executase dende terminal proporcionándolle diferentes parámetros. No noso caso os parámetros a utilizar serán:

- -t Tempo de captura de vídeo, no noso caso será 0 indicando que a captura será continua.
- -w e -h Son os parámetros de anchura e altura de píxeles, probaremos diferentes resolucións para conseguir a máxima calidade posible sempre que o tamaño da imaxe non repercuta na latencia de transmisión.
- -fps *Frames per second*, é o numero de imaxes a capturar cada segundo, variaremos con este valor para minimizar a latencia.
- -b *Bitrate*, o numero de bits por segundo, buscaremos o valor máis alto posible sen que produza retardos na transmisión.
- -n Con este parámetro deshabilitaremos a previsualización do vídeo.
- -pf Parámetro para elixir o perfil do codificador de vídeo H264, as opcións dispoñibles son, *baseline*, *main* and *high*. Utilizaremos a opción *baseline* xa que é a que menor custo computacional ten.
- -o Con este parámetro indicamos a saída de vídeo, como por exemplo a un arquivo, no noso caso utilizaremos a saída estándar que indicaremos con ”-” , e que redirecionaremos máis tarde.

4.5.3 Transmisión de vídeo

Para transmitir o vídeo ao dispositivo móbil a través da rede preséntanse varias posibilidades. Para comparalas transmitiremos vídeo dende a Raspberry Pi cunha mesma resolución, 720p, e o recibiremos e o reproduciremos nun pc mediante *vlc*.

- A primeira opción a analizar é o software de vídeo *vlc*, unha completa ferramenta de reprodución que tamén permite a transmisión e a recepción de vídeo na rede mediante diferentes protocolos. Fixose unha proba utilizando o *vlc* na Raspberry Pi para transmitir o vídeo, como resultado obteuse unha transmisión cunha latencia superior a un segundo, que imposibilita o seu uso para controlar o trafico en tempo real.

- A segunda opción a probar consistirá en capturar o vídeo coa ferramenta de captura de vídeo da Raspberry Pi, *raspivid*, e redirecionar a sua saída á rede utilizando *netcat* unha utilidade para transmitir e recibir na rede mediante *tcp* ou *udp*. Transmitiuse mediante *udp* para conseguir unha menor latencia a custo de perder algún fotograma. A latencia obtida neste caso e mellor que no anterior.

- Buscando reducir aínda máis a latencia probouse a utilizar o software *socat*, que funciona de forma similar a *netcat* e conta tamén con moitas opcións de configuración. O procedemento será igual que no caso anterior, faremos a captura con *raspivid* e redirecionaremos o vídeo a un porto nunha dirección *ip* mediante *udp* neste caso utilizando *socat*. Como resultado obtemos unha latencia aínda menor que con *netcat* polo que utilizarase este software para a transmisión de vídeo.

4.5.4 LipoPi

Este é un script Python encargado de comunicar a Raspberry Pi co circuíto de carga. A súa función é enviar unha orde de apagado o sistema que cerrará tódolos programas, incluíndo LightsServer e apagará ao sistema. Para isto rexistra unha función de *callback* que se executará cando se produce un cambio nun dos seguintes pins.

- O pin conectado á liña de baixo voltaxe do Adafruit Powerboost, que poñerase a 0 cando batería estea baixa.

- O pin conectado ao pulsador, que tomará un valor positivo cando se prema e cerre o circuíto.

Tamén xenera un arquivo de logs no que informará do motivo e o momento de cada apagado.

4.5.5 Servizos en *systemd*

Estes servizos executan un programa cando arranca o sistema e o reinicia cada vez que se pechen se así o indicamos. A súa estrutura é sinxela nela se indica a orde de execución a iniciar, e a ubicación do arquivo, tamén se lle poden indicar outros parámetros como se se quere que se reinicie sempre se se pecha, o tempo de espera en caso de reinicio do programa ou o momento de execución cando o sistema arranca. Para instalar un novo servizo se copiaría o arquivo *.service* a ubicación */etc/systemd/system/* despois se abilitaría coa orde

systemctl enable nomeDoServizo e se iniciaría coa orde *systemctl enable nomeDoServizo*

bikecam service

Este servizo executase co arranque do sistema e arranca ao servidor *LightsServer* despois de que se cargue as interfaces de rede.

lipopi service

Para arrancar automaticamente o script Python instalarase un novo servizo en *systemd* e de igual maneira que co servidor executarase no arranque e cada vez que se pare.

4.6 Carcasa e ancoraxe

Para protexer o dispositivo e suxeitalo baixo a sela da bicicleta proponse dúas opcións.

4.6.1 Versión con batería externa

A primeira realizarase para a versión do proxecto alimentada cunha batería USB. Consistirá en utilizar a carcasa oficial da Raspberry Pi Zero que inclúe un oco para a cámara e espazo para as conexión na parte de atrás, a carcasa so permite a uso de cámaras sen lentes polo que incorporarase unha lente externa.

Para suxeitar a carcasa á bicicleta deseñaremos un soporte en 3d co software Blender. Partiremos dás medición da carcasa e deseñarase un soporte que suxeite a carcasa firmemente e permita atala á barra da sela mediante unha correa.

Unha vez deseñado e tras comprobar que o deseño é imprimible exportarase no formato *STL* que abrirase cun software encargado de dividir o deseño en capas e traducilo a ordes de desprazamento interpretables pola impresora, aquí configurararanse diferentes parámetros como a altura de capa, que é a resolución de impresión, as velocidades, a cantidade e tipo de recheo da peza ou o uso de soportes para facilitar a impresión. Neste caso utilizaras o software libre Slic3r no que configurarase a peza a imprimir cun 100% de recheo para que sexa máis sólida cunha altura de capa de 0.2mm e sen uso de soportes, como resultado obterase un arquivo *GCODE* que pasaremos a impresora. O prototipo imprimirase o prototipo en 3d probarase e aplicaranse correccións no modelo. Para este deseño realizáronse dúas iteracións móstranse na figura 4.8.

4.6.2 Versión con batería interna

A segunda versión terá que albergar a Raspberry Pi Zero xunto coa cámara, o chip de carga e alimentación, o conversor lóxico de voltaxe e a batería. Utilizaremos tamén neste caso o software de edición 3d Blender para deseñar os prototipos.

O deseño contara co anel LED situado no exterior ao redor da lente da cámara, no interior colocaranse tódolos compoñentes electrónicas e a batería. Na parte superior contará cun oco para o conector micro USB de carga e aceso á tarxeta micro sd da Raspberry Pi. No exterior contará tamén con dous brazos articulados nos que situaremos as dúas tiras LEDs para indicar o xiro.

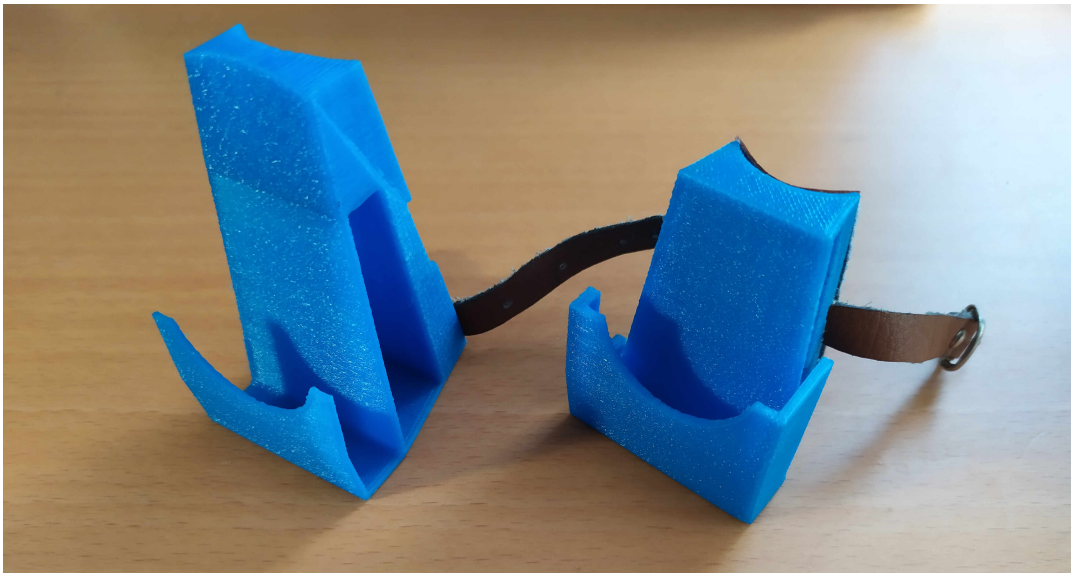

Figura 4.8: Deseños do soporte na bicicleta para a caixa oficia da Rasberry Pi.

Da mesma forma que no caso anterior o deseño imprimirase modificarase e volverase a imprimir ata que se obteña un resultado aceptable. Na figura **??** pódese observar a evolución dos diferentes deseños e na figura 4.9 unha imaxe renderizada do deseño final.

Figura 4.9: Evolución dos deseños da carcasa.

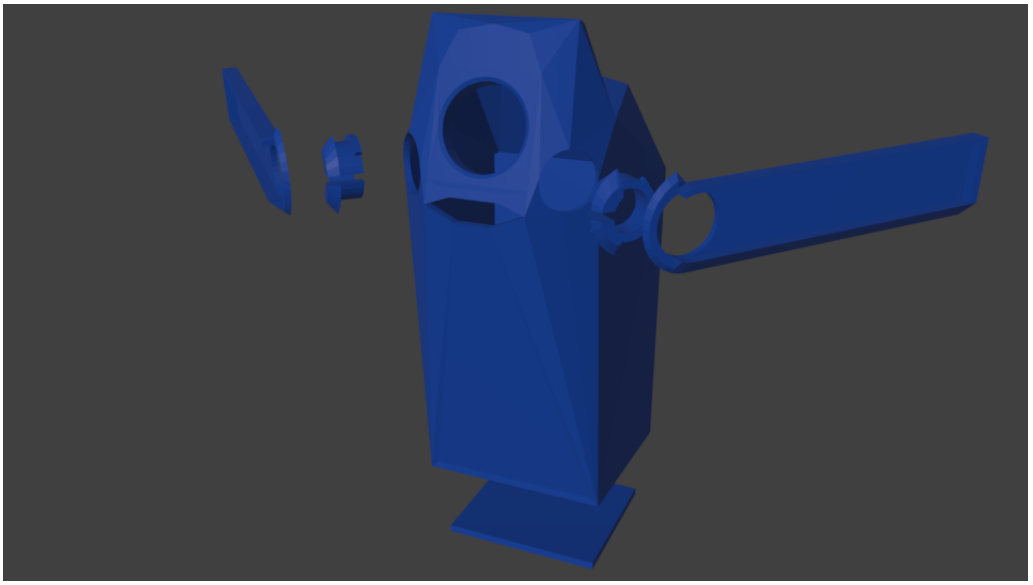

Figura 4.10: Ultimo deseño da carcasa.

Capítulo 5

Deseño e implementación de BikeView

Partindo dos requisitos de BikeView elixirase a estrutura xeral da aplicación Android. A continuación mostrarase a actividade principal e única da aplicación e explicarase a implementación das comunicacións e da recepción do vídeo H264. Rematarase comentando diversos tipos de ancoraxes de BikeView á bicicleta.

5.1 Esquema xeral da aplicación

Para crear esta aplicación Android optarase por utilizar a linguaxe de programación Kotlin e o entorno de programación Android Studio. Kotlin é unha linguaxe de programación creada por JetBrains executable na maquina virtual de Java, dende 2017 Kotlin é unha linguaxe oficial para desenvolver aplicacións Android.

Como mosta o libro Kotlin for Android Developers [14] algunhas das súas vantaxes fronte a Java son:

- Unha maior expresividade: Podes escribir máis con menos código.
- Maior seguridade: En Kotlin e obrigatorio especificar a nulabilidade dos obxectos e esta se comproba en tempo de compilación.
- É funcional: Kotlin é unha linguaxe orientada a obxectos pero inclúe conceptos da programación funcional como as expresións lambda.
- Fai uso de extensión de funcións: Permite estender clases con novas funcionalidades se necesidade de ter aceso o código da clase.
- É altamente interoperable: Pódese utilizar librerías e clases de Java no mesmo proxecto.

A aplicación contará con catro compoñentes. O principal será a *MainActivity*, a encargada de iniciar as compoñentes a visualizar na pantalla, executar as ordes e mostrar a información. O segundo será o *layout* onde se definiran as compoñentes a visualizar e a sua posición en pantalla. O terceiro a clase *request* a encargada da conexión co servidor e de transmitirlle as ordes. O cuarto elemento é a clase encargada de recibir o vídeo e descodificalo.

5.2 Actividade principal

A actividade principal ou *MainActivity* dunha aplicación Android é a primeira pantalla que aparece cando executase a aplicación e a encargada de controlar o seu funcionamento e a sua interface de usuario. Neste caso a actividade será a encargada de mostrar os botóns e encargarse do seu funcionamento, amosar o estado da conexión, encargarse da xestión do sensores e reproducir o vídeo. O mesmo tempo encargarase de instanciar e comunicarse coas clases encargadas da conexión co dispositivo e da recepción de vídeo.

5.2.1 Layout

Un *layout* é a definición da estrutura da interface de usuario. Esta actividade contará con dous *layouts* un para a posición vertical e outro para a posición horizontal da pantalla

Layout vertical

Este *layout*, figura 5.1, contará cunha superficie reservada para o vídeo na metade superior da pantalla. Na metade inferior situaranse os botóns encargados de executar as ordes. Na parte superior situase a barra de estado que na sua parte dereita contará cun botón para conectar que servirá o tempo de indicador de conexión, a súa esquerda mostrarase unha barra para controlar a intensidade dos LEDs.

Layout horizontal

Neste *layout*, figura 5.2, o vídeo mostrarase como o fondo e os botóns sobre el. Os de xiro a esquerda e xiro dereita situados a ámbolos dous lados e o resto incluíndo o de control de intensidade lumínica e o de conexión situaranse na parte de inferior da pantalla.

5.2.2 Ciclo de vida da actividade

En Android cada actividade conta conta cun ciclo de vida [15], pasa por varios estados dende antes de iniciarse ata despois de finalizar. O estado principal dunha actividade é activa, no momento que a actividade está en primeiro plano e interactuando co usuario.

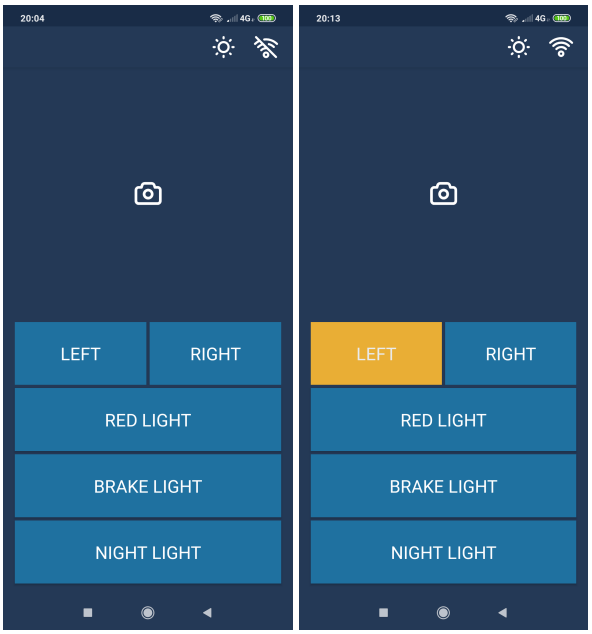

Figura 5.1: Capturas de pantalla do *layout* horizontal.

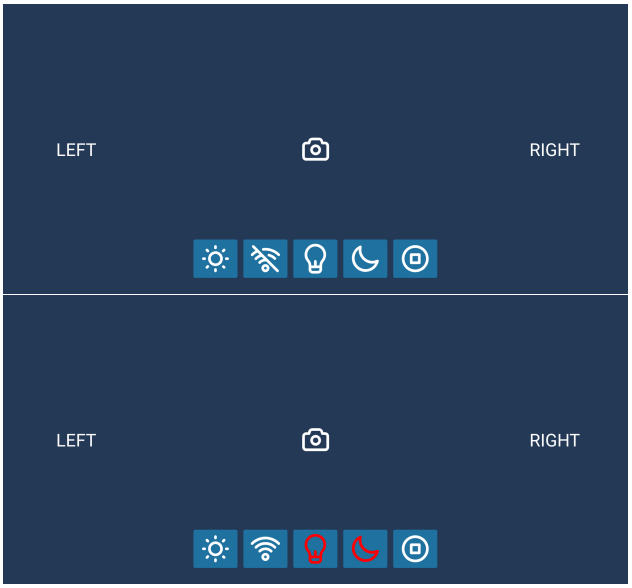

Figura 5.2: Capturas de pantalla do *layout* vertical.

Para xestionar o que sucede no resto de estados utilízanse unhas funcións de *callbacks*, no noso caso realizarase as seguintes accións en cada fase:

onCreate

Este método e o que se chama ao executar a actividade. Nel iniciarase os compoñentes a mostrar en pantalla e definirase o seu comportamento.

Aquí xestionarase o estado da conexión, iniciando a clase *request* se non esta en funcionamento e enviando mensaxes o dispositivo para comprobar que segue conectado. Tamén xestionarase o estado da transmisión de vídeo xa que será necesario iniciar a recepción antes de iniciar a transmisión.

onResume

Este método executase despois de *onCreate* cando a pantalla xa é visible para o usuario.

Rexistrarase aquí un *sensor listener* para obter a información do sensor de luz.

onPause

Este método executase cando a aplicación deixa de estar en primeiro plano, se o usuario a minimiza u outra aplicación executase a actividade pasará a *onStop* se despois de acceder a aplicacións recentes a aplicación volve a primeiro plano pásese o método *onResume*.

Levarase a cabo neste método a cancelación do rexistro do sensor xa que non se seguira a utilizar e deterase a transmisión de vídeo se estase a executar.

onDestroy

Neste método a actividade e detida completamente e devénse liberar os recursos.

Aquí enviarase a orde para deter a conexión co dispositivo.

5.2.3 Botons

A actividade é a encargada de iniciar os botóns e manexar o seu funcionamento. Estes botón son os seguintes:

- **Esquerda:** O pulsar este botón enviarase unha orde de acender a luz de xiro a esquerda. Se hai algunha luz acesa se enviará primeiro unha orde para apagalas. O pulsalo por segunda vez se enviara a orde de apagar luz de xiro e no caso de que a luz vermella estivese acesa antes de indicar o xiro esta luz se acenderá de novo. O botón palpebrará en amarelo mentres estea aceso.
- **Dereita:** O seu funcionamento é o mesmo que no botón esquerda.

- **Vermello:** O pulsalo enviarase unha orde para acender a luz de vermella, o pulsador non funcionara se algunha das luces de xiro está acesa.
- **Noite:** O pulsalo activarase o modo noite, no que o sensor lumínico do móbil encargarase de acender a luz vermella cando a luz ambiente baixe de certo umbral.
- **Freo:** Activa o modo de freada no que acenderase a luz vermella progresivamente cando o acelerómetro do dispositivo móbil detecte unha redución súbita na velocidade.
- **Brillo:** Este botón despregará unha barra cun indicador que poderase deslizar para elixir a intensidade das luces.
- **Conexión:** Este botón enviará unha orde de conexión, unha vez conectado cambiará a súa aparencia para indicar que existe conexión co dispositivo.

5.2.4 Sensores

Unhas das vantaxes de utilizar un dispositivo móbil é que estes contan con diversos sensores para moitos fins. No noso caso utilizarase o acelerómetro e o xiroscopio para rexistrar cambios na posición e aceleracións no dispositivo e o sensor de luz para medir a intensidade de luz no ambiente. En Android accedese o sensor instanciando a clase *SensorManager* e definindo unha instancia do sensor da que obterase os datos mediante a función *on sensor change* [16].

Sensor de Luz

O sensor de luz é un fotorreceptor que xera unha sinal eléctrica dependendo da incidencia de fotóns. Para utilizar o sensor rexistrarase un *sensor listener* no método *onResume* e o método de *callback onSensorChanged* executarase cada vez que se detecte un cambio, neste método definirase o comportamento do sensor: Cando o botón de Noite esta activado acenderase o botón Vermello se o sensor rexistra un valor inferior a 400 lúmenes. Este valor correspondese coa intensidade lumínica o comenzo do solpor [17].

Acelerómetro e xiroscopio

Para poder indicar a freada da bicicleta utilizarase o acelerómetro do dispositivo co que medirase a deceleración. O acelerómetro proporciona o valor da aceleración en m/s^2, para simplificar os cálculos utilizarase o sensor virtual *linear accelerometer* que proporciona o valor da aceleración excluíndo a gravidade asi o valor será 0 nos tres eixos cando o dispositivo este en repouso ou movéndose a unha velocidade constante.

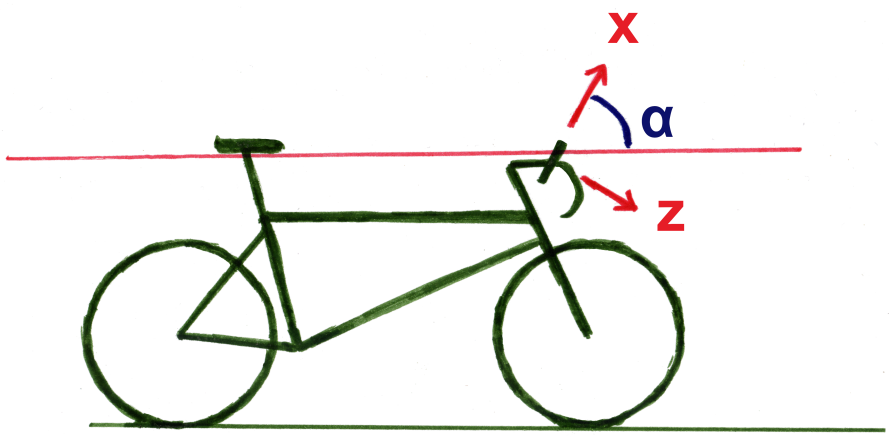

Figura 5.3: Eixos do acelerómetro e ángulo coa horizontal. Para o dispositivo colocado en horizontal, en vertical o eixo superior sería y.

A forza a obter é a deceleración no sentido da marcha, para poder calculala é necesario coñecer a orientación do dispositivo co respecto ao eixo lonxitudinal da bicicleta. O dispositivo situarase no guiador da bicicleta en dúas posición horizontal, ou vertical polo que en ámbalas dúas poderase excluír os valores dun dos eixos do acelerómetro, x e y respectivamente como se ve na figur 5.3. Para coñecer a influenza dos outros dous eixos sobre o plano paralelo á bicicleta necesitamos coñecer o ángulo que forma con esta. Utilizaremos o sensor virtual *game rotation vector* para obter este ángulo, este sensor utiliza os datos do acelerómetro e o xiroscopio para proporcionar valores rotación en radiáns sobre os eixo de coordenadas da terra, aínda que non orientado o norte xa que este sensor virtual non utiliza o magnetómetro. Estes valores están expresados en ángulos de Euler que indican mediante unha sucesións de xiros a desviación do sistema de coordenadas, para obter o ángulo respecto o eixo de coordenadas do móbil transformaremos o vector obtido mediante o calculo a matriz de rotación. Unha vez obtidos valores normalizados utilizaremos o ángulo respecto o eixo x se o móbil esta en posición vertical e con respecto a y se está en horizontal.

O ángulo obtido coincide co ángulo con respecto o eixo direccional da bicicleta so no caso no que á bicicleta estea sobre chan plano polo que será necesario que á bicicleta estea sobre un terreo sen inclinación, para elo o cálculo do ángulo realizarase nun proceso de calibración, cando se acenda o modo freada solicitarase que se coloque o telefono no soporte do guiador e se suxeite á bicicleta sobre un terreo chairo.

O modulo da forza resultante calcularase de forma trigonométrica a partir dos valores de aceleración dos dous eixos e o ángulo obtido. O sistema indicará que se acenda as luces cando

o valor calculado supere certo nivel, utilizarase en principio o valor de m/s^2 e comprobarase nas probas se é o valor adecuado.

De igual forma que no sensor de luz estes sensores rexistraranse no método *onResume* e os cálculos cos valores obtidos realizaranse no método *onSensorChanged*.

5.2.5 Superficie de vídeo

Iniciarase a superficie de vídeo na fase de creación da aplicación, procederase a reflectir a superficie para facer un efecto de espello para facer máis natural a a visualización do vídeo. O vídeo asignarase a superficie iniciando a clase *VideoReceiver* mediante as funcións de *callback* que se executarán cando se produza un cambio na superficie de vídeo.

5.3 Comuniciacíon co dispositivo

Crearase unha clase *Request* encargada de tódalas comunicacións co dispositivo. Esta clase, unha vez instanciada, encargarase de establecer a conexión co dispositivo, reconectar se se perde a conexión e transmitirlle as ordes.

Esta clase se instanciará na actividade principal e para que as transmisións e operacións de rede, como asignación de *sockets* ou transmisións de ordes e recepción de respostas poidan realizarse cocurrentemente utilizaranse as corrutinas de Kotlin que permiten facer estas chamadas de forma secuencial pero estas executaranse en diferentes *thread*.

5.3.1 Broadcast

Para coñecer a dirección IP do servidor, a clase *request* conta cunha función que abrirá un socket no porto 5555 para esperar a recepción dun datagrama broadcasteado a tódalas direccións. O recibir o datagrama comprobase que conten a mensaxe *"BikeView"* se é asi a función devolverá a dirección IP emisora do paquete.

5.3.2 Conexión

Unha vez obtida a dirección IP a clase *request* utilizará unha función para establecer a conexión que devolverá un socket conectado ao socket remoto do servidor.

5.3.3 Transmisión de ordes

Para transmitir as ordes a clase *request* contara cunha función que recibe a mensaxe a transmitir e a envía a través do socket. Espera a recibir resposta do servidor e se é positiva devolve o valor booleano verdadeiro, en caso de non recibila devolve o valor falso.

5.4 Recepción de vídeo

Para este apartado crearase a clase *VideoReceiver* que executarase no seu propio *thread*. O vídeo é codificado no formato H.264 e transmitido por UDP, para recibilo ábrese un *socket* no porto 5000 extraese a mensaxe do datragrama e, para acelerar o proceso, en vez de utilizar a función *MediaExtractor* pásase a unha función de parseo [18] encargada de separar os datos recibidos en *NAL units* [19], o formato que utiliza H.264 para o transporte. Para elo cando se identifica a cabeceira, o código *0x000001* da seguinte *NAL units* a anterior se envía a outra función encargada de decodificar o vídeo mediante un *MediaCodec* [20], finalmente o vídeo mostrase na superficie.

5.5 Ancoraxe á bicicleta

Existen diversas opcións paras suxeitar o móbil o guiador dunha bicicleta, no mercado hai soportes para dispositivos concretos e outros que se adaptan ao tamaño e forma de diferentes modelos. Neste caso utilizarase un código [21] *SCAD* que ao executalo co software de deseño 3D paramétrico OpenSCAD no que introdúcense as dimensións do dispositivo e como resultado obtense un arquivo *STL* co deseño 3D do soporte, figura 5.4. Vendo que os soportes impresos non constaban coa resistencia esperada, como se mostra nas imaxes da figura 5.4, optouse por utilizar un soporte de aluminio nas probas para preservar a integridade do dispositivo móbil.

Figura 5.4: Mostras dos problemas do soporte impreso.

Capítulo 6

Evaluación experimental

PARA comprobar que os resultados obtidos correspóndense co plantexado neste capítulo probaremos o dispositivo BikeCam e a aplicación BikeView. Realizaranse probas en contornos controlados probas no medio real o que esta dirixido. Centrarémonos na análise do consumo e a autonomía, a calidade das imaxes capturadas e a visibilidade das luces. Finalmente analizaramos se cumpre os requisitos legais.

6.1 Consumo

Comenzarase as probas medindo o consumo de amperios do sistema. Para elo colocarase un amperímetro USB entre unha fonte de alimentación e a conexión USB coa que alimentarase o sistema nestas probas. Repetiremos as probas con diferentes fontes de alimentación e distintos cables para descartar fallos e conseguir unha maior consistencia nos resultados. A continuación realizaranse as mesmas probas no dispositivo medindo o consumo alimentándoo coa batería.

Analizaranse os seguintes supostos para o dispositivo con un anel LED e para o que conta a maiores coas dúas tiras LED.

- **Sistema en repouso:** O sistema esta aceso pero só se estean a executar as funcións do sistema operativo incluíndo o servidor ssh para o control remoto.
- **Servidor funcionando:** Executamos o servidor.
- **Cliente conectado:** Conectamos o dispositivo móbil o servidor.
- **Vídeo transmitindo:** Transmitimos vídeo en directo o dispositivo móbil.
- **Vídeo parado:** Paramos a transmisión de vídeo.
- **Desconexión do cliente:** Pechamos a aplicación no dispositivo móbil.

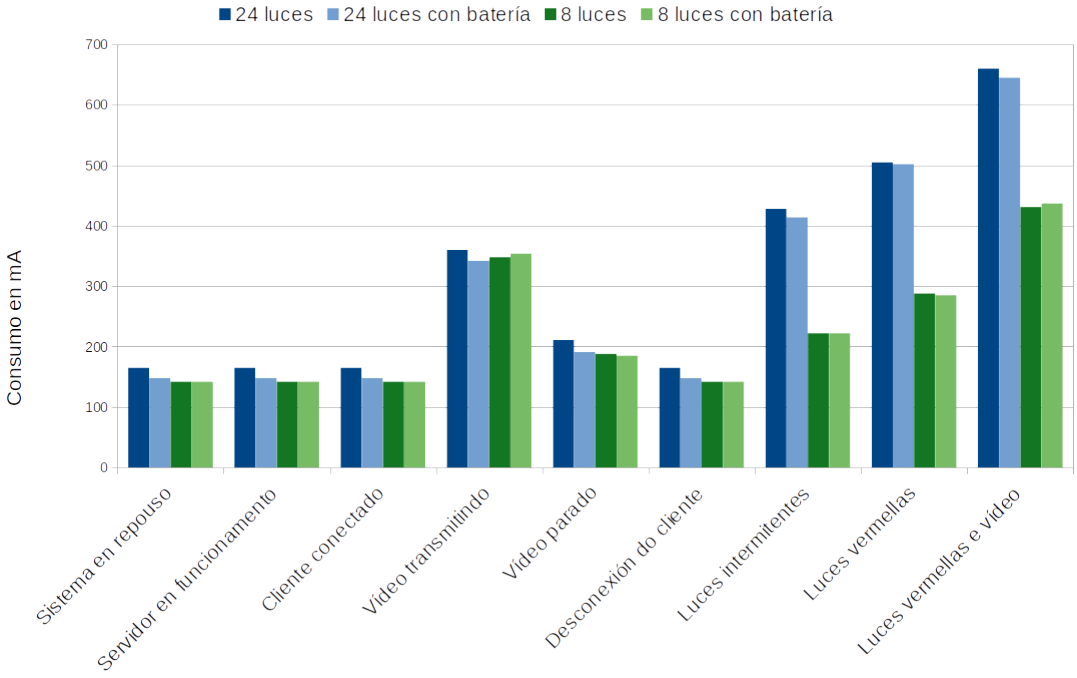

Figura 6.1: Grafica do consumo en mA.

- **Luces intermitentes:** Iniciamos as luces intermitentes a máxima intensidade nunha das direccións, consumo varia no proceso, rexistremos o valor máximo.
- **Luces vermellas:** Acendemos as luces vermellas a intensidade máxima.
- **Luces vermellas e transmisión de vídeo:** Consumo coas luces vermellas a máxima intensidade e co vídeo transmitindo en directo.

Como se pode observar na gráfica da figura 6.1 o uso ou non de batería non afecta en gran medida ao consumo de amperios, pero ha de terse en conta que estas probas realizáronse coas batería coa carga completa. Se se compara a previsión de consumos feita coas medición obtidas pódese ver que os valores no son moi diferentes sendo superior no caso do dispositivo de 8 luces, 437mA fronte ós 390mA previstos, e inferior no caso dispositivo con 24 luces, 660mA fronte ós 710mA previstos. As fontes de alimentación utilizadas poden proporcionar un máximo de 3A e 2.1A respectivamente, tamén utilizáronse dous cables un de maior calidade e outro cunha calidade inferior. En ningún dos casos obtivéronse diferencias apreciables sendo a maior diferencia de 4mA. Na versión con batería necesitouse utilizar cableado a maiores para realizar as probas, o que pode que incrementara lixeiramente o consumo.

6.2 Autonomía

A seguinte proba a realizar será a de autonomía do dispositivo, para elo buscarase o consumo máximo acendendo as luces vermella a máxima intensidade o tempo que se transmite vídeo.

Comezaranse as probas co dispositivo dotado de batería interna, cando a voltaxe da batería esta entorno os 3.9V detectase un problema o sistema apagase, indicando batería baixa. O *script* Python encargado de monitorizar o pin conectado o indicador de voltaxe baixo do Adafruit Powerboost detecta unha caída de voltaxe que confunde co franco de caída esperado cando o pin ponse a 0. O pin de baixo voltaxe do Adafruit Powerboost está conectado a voltaxe da batería cando a caga e superior a 3.2V e conectase a un valor de 0V cando baixa de este limite e debido a que cando batería esta a comezar a descargarse para poder proporcionar a amperaxe adecuada a súa voltaxe baixa a un ritmo rápido que confunde o *script*. O podemos solucionar incluíndo unha segunda comprobación no *script* despois de detectar o franco de caída comprobarase que o valor lido no pin é 0 antes de apagar a Raspberry.

A continuación detectase un segundo problema, despois de 15 min de funcionamento volvese a apagar por batería baixa e unha vez apagado a batería recuperase ata unha voltaxe de 3.6V. Esto debese a que o estar consumindo o máximo de amperios necesita baixar a súa voltaxe para poder seguir entregando estes amperios. Reducindo a intensidade dos LEDs conséguese a que o sistema non se apague pero a medida que a voltaxe vai baixando é necesario reducir a intensidade aínda máis. Este problema pódese solucionar de dúas formas: Utilizando un batería de maior capacidade xa que poderá seguir subministrando a amperaxe necesaria a menor voltaxe. Utilizar unha batería cunha constante de descarga maior, isto é a capacidade máxima de amperios que pode subministrar en función da capacidade en amperios hora. A batería utilizada ten unha constante de descarga de 1C isto quere dicir que coa sua caga completa o fabricante garante que proporcione 1.6A funcionando con normalidade. Ámbalas dúas solucións implican na practica unha batería de maior tamaño.

Procedemos a repetir a proba pero esta vez deshabitando o apagado automático, esperando a que o sistema se apague cando a voltaxe non sexa suficiente para manter a Raspberry acesa ou no peor caso cando o circuíto de protección que inclúe a batería a desconecte a unha voltaxe mínima habitualmente 2.5V. De non usar unha batería con circuíto de protección poderíamos danar a batería podendo incluso arder ou estoupar durante próximas cargas. Neste caso obtivéronse 35 minutos de funcionamento. A voltaxe medida na batería foi de 3.58V, un valor de voltaxe na que a batería aínda conserva parte da súa carga. Que o sistema se apague neste punto e conveniente para protexer a batería e prolongar a súa vida útil pero reduce o tempo de uso da batería, que poderíamos seguir usando sempre que non utilicemos o consumo máximo do dispositivo, por exemplo apagando a transmisión de vídeo. Comprobamos

Figura 6.2: Capturas feitas coa cámara V1, V2, e V2 sen filtro infravermello.

este suposto reconectando a batería e esta vez acendendo so as luces e a unha intensidade do 50%, obtivéronse neste caso 50 minutos máis de funcionamento ata que a batería se desconectou mediante o seu circuíto de protección, para que volva a activarse e necesario comezar a recargala, a voltaxe medida foi de 3.4V.

Comezarase as probas para a versión do dispositivo sen luces intermitentes e alimentando por batería USB, faremos a proba cunha batería de 2600mA, despois de 3h e 30min e coa batería a unha voltaxe de 3.2V o vídeo deixa de funcionar pero os LEDs seguen a responder o pouco tempo o control dos LEDs tamén deixa de funcionar e a Raspberry se apaga pero as luces segue acesas se é esta a posición na que as deixamos, o chegar a 4 horas de funcionamento das luces e unha voltaxe de 2.9V desconectamos o sistema, e o intentalo arrancar de novo a voltaxe no e suficiente para arrancar a Raspberry.

Durante esta proba tamén se monitorizou o consumo enerxético do dispositivo android executando BikeView, para esta proba utilizouse un dispositivo cunha pantalla de 6,3" unha batería de 4000mAh e co brillo de pantalla a metade de intensidade. O comezo da proba o dispositivo contaba co 100% da batería o rematar a proba despois de 4h a batería restante era do 64%.

6.3 Vídeo e lentes

Como se pode observar nas capturas da figura 6.2, a calidade de imaxe é considerablemente maior cando se utiliza a versión 2 da cámara, esta diferenza é aínda máis notable cando as condicións de iluminación son desfavorables. Na 6.3 podemos observar captura de imaxe realizadas coa versión 1 da cámara e tres tipos de lente distintas que se enroscan na propia cámara. As lentes contan con 130º 180º e 220º. Co se pode observar a distorsión na lente de 220º é demasiado grande,o uso de calquera das outras dúas sería apropiado para captura a amplitude da vía.

Para medir a latencia gardaremos a captura de vídeo do dispositivo BikeCam gravando un cronómetro o tempo que grava a pantalla do dispositivo BikeView. Analizando posteriormente o vídeo calcúlase a latencia parando o vídeo e restando o valor do cronometro visualizado

Figura 6.3: Comparativa de lentes de 130º 180º e 220º

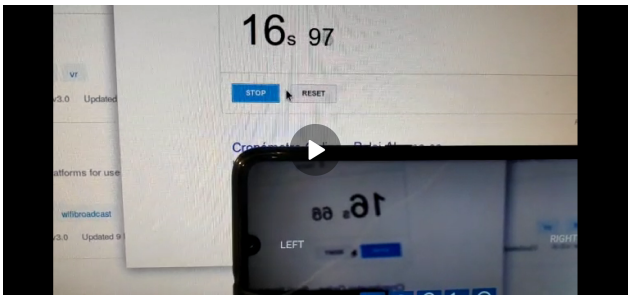

Figura 6.4: Medición empírica da latencia.

da pantalla menos o valor real do cronómetro como se aprecia na figura **??**. Repetindo o procedemento en varios puntos do vídeo observamos unha latencia constante de entorno a 320ms para as probas realizadas coa versión un da cámara e 295ms para os probas coa versión 2. Cabe destacar que na proba de autonomía realizada no dispositivo BikeCam con 24 LEDs cando este se encontraba a un nivel baixo de batería e un consumo máximo, *streaming* de vídeo e máxima intensidade de luces, apreciouse nalgún momento subidas repentinas de latencia e defecto na imaxe seguramente devidos a faios de codificación.

6.4 Visibilidade

Para comprobar a visibilidade das luces procedeuse a observar o seu funcionamento a diferentes intensidades e distancias repetindo as probas de día e de noite. Sendo as luces amarelas máis brillantes e o padrón de movemento máis visible decidiuse facer as probas só nas luces vermellas e sen intermitencia para ilustra o caso menos visible.

As distancias elixidas para a proba foron 50, 100 e 150 metros e as intensidades elixidas 20, 40, 80 e 100 por cento. Nas visualizacións realizadas de día a luz vermella era distinguible en todas as distancias e non se apreciaron diferenzas entres as distintas intensidades, isto pode deberse a que as intensidades utilizadas seguen unha escala lineal mentres que a percepción visual do ser humano se rexe por unha escala logarítmica, figura 6.4, podería cambiarse a escala da regulación de intensidade por unha logarítmica para conseguir unha representación

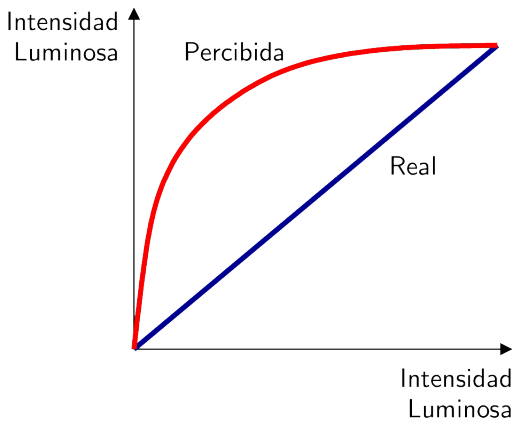

Figura 6.5: Perecepción subxectiva de estímulos lumínicos segundo a lei de Weber-Fechner [1].

máis realista.

Nas visualizacións realizadas de noite as luces se percibía mellor que de día e neste caso si se podía distinguir as intensidades máis baixas das máis altas. Na figura 6.5 móstranse unhas fotos da luz a intensidade máxima a 150m de día e de noite. Na figura 6.6 mostrase a luz a 80% a 150m comparada coas luces de freada dun coche a mesma distancia.

6.4.1 Modo noite

Para comprobar o correcto funcionamento destes dous modos situouse o telefono no guiador da bicicleta e se realizaron as seguintes probas. Comprobouse que no modo noite a luz vermella se acendía automaticamente ao entra nun túnel e se apagaba o saír de este, tamén se comprobou que se acendía correctamente ao solpor. Se ben o valor de luminosidade foi o adecuado podería ser conveniente que o usuario puidera controlar este limiar de luminosidade mínima.

6.4.2 Freo automático

As probas de deceleración realizáronse freando coa bicicleta en vías con diferente inclinación e co dispositivo Android tanto en posición horizontal como posición vertical. As luces acendéronse correctamente cando a bicecleta realia unha redución de velociade. De igual forma que co sensor de luminosidade seria conveniente que o usuario puidera establece o mínimo limiar de forza de deceleración a que quere que se acenda as luces.

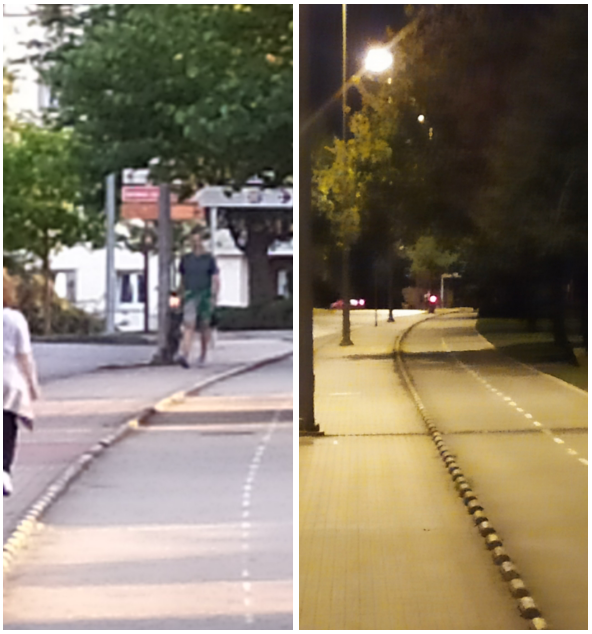

Figura 6.6: Fotos (recortadas) da luz vermella a 150m. A esquerda de día a dereita de noite.

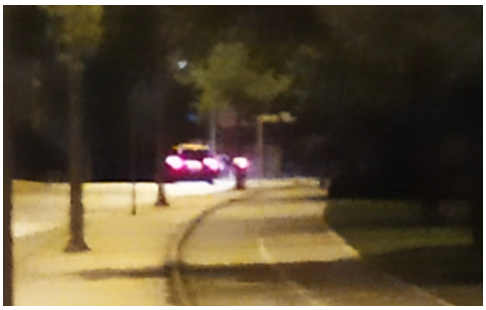

Figura 6.7: Comparación das luces vermellas xunto as luces de freada dun coche a 150m.

Intensidad luminosa (candelas)	Delanteros (1)	4-60
	Traseros (1)	4-12

(1) en la dirección del eje.

Figura 6.8: Rango de intensidade lumínica nos faros das biciletas en España [2].

6.5 Lexislación

A Lexislación que regula o uso de luces e cámaras varía amplamente segundo o país, analizaremos a legalidade do dispositivo segundo a lexislación española.

6.5.1 Luces

O uso de luces en bicicleta e obrigatorio en España cando as condicións lúminicas son insuficientes, na Instrución 18/S-146 da Dirección Xeneral de Tráfico con respecto a legalidade de luces parpadeantes, detalla os requisitos lumínicos aplicables as bicicletas recollidos nas diferentes lexislacións [2]:

- Artigo 98.1 do Regulmento Xeneral de Circulación, aprobado por Real Decreto 1428/2003, de 21 de novembro.
  ”Todos los vehículos que circulen entre el ocaso y la salida del sol o a cualquier hora del día en los túneles, pasos inferiores y tramos de vía afectados por la señal “túnel” (S-5) deben llevar encendido el alumbrado que corresponda de acuerdo con los que se determina en esta sección”.

- Regulamento Xeneral de Vehículos, aprobado por Real Decreto 2822/1998, de 23 decembro
  ”En el artículo 22.4 se indica que las bicicletas, para circular de noche, por tramos de vías señalizados con la señal de “túnel” o cuando existan condiciones meteorológicas o ambientales que disminuyan sensiblemente la visibilidad, deberán disponer de los siguientes dispositivos: luz de posición delantera y trasera, catadióptrico trasero, y podrán disponer de catadióptricos en los radios de las ruedas y en los pedales”.

- Real Decreto 339/2014, de 9 de mayo, que establece os requisitos para a comercialización e posta en servizo das bicicletas e outros ciclos e as súas partes e pezas,establece no Anexo V o rango de intensidade lumínica nos faros das bicicletas (ver figura 6.7).

A instrución conclúe que o uso de luces parpadeantes é legal sempre que as luces no produzan cegamento o resto de usuarios da vía e se adecúen a normativa citada. Para saber se

as luces cumpren estes requisitos lumínicos tense que calcular as candelas que emiten. Os fabricantes de tiras LEDs raramente indican esta información e se o fan é en formato de lúmenes que para converter a candelas e necesario saber o ángulo de emisión, dato que tampouco facilitan, e non tódolos LEDs co mesmo formato compórtanse da mesma maneira. Na ficha técnica duns LEDs WS2812B, coma os utilizados neste proxecto, do fabricante Worldsemi [22] na táboa 6.1 especifican as milicandelas dos seus LEDs, tendo unha intensidade lumínica máxima combinada de entre 1850 e 2500 milicandelas.

O dispositivo desenvolto so utiliza dúas cores a vermella cun valor *RGB* (255,0,0) e a amarela cun valor *RGB* (255,170,0). Polo tanto para a luz vermella solo utilizamos este LED cunha intensidade máxima indicada na táboa deste fabricante de entre 550 en 700 milicandelas. Para a luz amarela o LED vermello funciona a intensidade máxima e o verde cun valor de 177 dun máximo de 255, supoñendo unha transmisión lineal destes valores a intensidade en candelas, obteríase un valor do 69.4% que sobre os datos do fabricante para o LED verde resulta entre 763 e 961 milicandelas, sumados os valores do LED vermello o resultado é un máximo de entre 1313 e 1661 milicandelas.

Para a luz vermella, no dispositivo de 8 luces isto serán entre 4.4 e 5.6 candelas, dentro do rango legal de entre 4 e 12 candelas, no caso do dispositivo de 24 luces o intensidade máxima será de entre 13.2 e 16.8 candelas en principio superior o limite legal non obstante este requirimento de intensidade é na dirección do eixo e as 16 luces extras utilizadas neste dispositivo están colocadas nun ángulo duns 55 graos polo que a intensidade na dirección será inferior, non e posible calcular este valor sen coñecer o ángulo de emisión de luz de estes LEDs. Para a luz amarela entre 10.5 e 13.2 candelas, podendo ser neste caso superior o limite legal, no dispositivo de 24 luces o valor máximo será de entre 31.5 e 39.8 candelas aínda que de igual forma que as vermellas o valor na dirección do eixo será menor.

Como se pode observar en tódolos casos, a priori, as luces acadan o limite legal pero nalgúns o sobrepasan, para adaptar o comportamento das luces a lexislación bastará con limitar o valor da intensidade máxima. Para poder obter este valor será necesario realizar probas empíricas para medir as candelas emitidas polo dispositivo.

Cabe destacar que o uso de indicadores de xiro non está recollido na lexislación española polo que seu uso podería ser sancionable si se considera que pode ser contraproducente cegando a visión doutros ocupantes da vía. O seu uso non eximirá a necesidade de indicar as manobras co brazo como require a lei.

6.5.2 Cámaras

Como o dispositivo non almacena as imaxes capturadas nin as comparte con terceiros non se esta a vulnerar ningunha lexislación. De incorporar a opción de gravar a vía poderíase incorrer nun delito nalgúns casos, xa que non existe unha regulación concreta sobre este

Táboa 6.1: Características dos LEDs WS2812B de Worldsemi

Cor a emitir	Modelo	Lonxitude de onda (nm)	Intensidade lumínica (mcd)	Voltaxe(V)
Vermello	13CBAUP	620-630	550-700	1.8-2.2
Verde	13CBAUP	620-630	1100-1400	3.0-3.2
Azul	10R1MUX	465-475	200-400	3.0-3.4

tipo de cámaras. Non será legal gravar a vía publica se se realiza de forma continuada e co vehículo parado xa que se pode considerar vídeo vixilancia ilegal, pero poderase capturar imaxes en movemento sempre que a utilización de estas se restrinxa o uso privado xa que a compartición de imaxes de persoas e das matriculas dos vehículos na vía pública vulnera a Ley Orgánica de Protección de Datos de Carácter Personal (LOPD) [23] sempre que non se conte co consentimento destas persoas. En caso de accidente as imaxes poderían utilizarse nun xuízo ou nunha reclamación o seguro só con autorización xudicial.

Capítulo 7

Conclusións e traballo futuro

Neste derradeiro capítulo preséntanse as conclusión do proxecto relatando o proceso seguido e analizando os resultados obtidos. Finalmente na sección de traballo futuro abórdanse as posibles opcións de melloras e expansións.

7.1 Conclusións

Aumentar a seguridade dos desprazamentos en bicicleta é fundamental á hora de afrontar os cambios nas novas sociedades urbanas. O obxectivo deste traballo foi o estudo, deseño, implementación e posterior avaliación dun sistema composto por cámara e luces para mellorar a seguridade dos ciclistas na vía publica. Para elo establecéronse uns requisitos funcionais, indicar a posición e manobras do ciclista e permitirlle visualizar o que ocorre as súas costas. Uns requisitos de deseño, pequeno tamaño, portabilidade, baixo custo e modularidade. Esbozouse un sistema composto por luces, cámara e pantalla, analizáronse as posibles opcións de implementación e tecnoloxías a utilizar e deseñouse un sistema composto por dous dispositivos: BikeCam, un miniordenador Raspberry Pi con luces RGB e cámara conectado mediante Wi-Fi a BikeView, unha aplicación de control e visualización funcionando nun dispositivo Android. Desenvolveuse un prototipo do sistema respectando os requisitos orixinais e tratouse de optimizar o seu funcionamento e usabilidade poñendo atención ós padróns lumínicos a independencia enerxética e a baixa latencia do vídeo. Por último púxose o sistema a proba para comprobar que o seu funcionamento e a súa usabilidade se correspondían o esperado.

Con estas probas demostrouse que o dispositivo é completamente funcional se ben conta con moitos aspectos a mellorar se quérese estandarizar o seu uso. Estas melloras implicarían o deseño dunha mellor caixa protectora que soporte os impactos e as condicións meteorolóxicas, a redución da latencia mellorando a decodificación, a mellora da robustez nas comunicacións para garantir a modularidade e facilitar implementación de novos casos de uso, engadir opcións de configuración para darlle ao usuario un maior control do sistema, ou o uso de

difusores nas luces para mellorar a visibilidade e protexer os LEDs.

7.2 Traballo futuro

O uso de compoñentes xenéricos e potentes pode permitir a evolución do dispositivo de diferentes formas, distinguiremos as seguintes:

- Mellorar as funcións actuais incluído un sistema de carga rápida da batería, un modo baixo consumo para aforrar batería se o dispositivo non está en uso ou realizar un circuíto impreso para simplificar a construción do dispositivo BikeCam.
- Engadir funcionalidades novas ao sistema como podería ser un detector de caídas, a gravación de vídeo, o cálculo da velocidade por GPS, ou o posproceseado do vídeo xa sexa en BikeCam (p. ex. TensorFlow) ou no dispositivo Android (p. ex. ARCore) o que permitiría novas funcionalidades como a detección de proximidade ou cambio de carril, para informar o ciclista do que sucede no seu entorno ou para advertir a outros condutores.
- Complementar o sistema con luces frontais e laterais e con mecanismos físicos de control ou control por voz.
- Diversificar os casos de uso como podería ser o uso de varios dispositivos ao mesmo tempo para controlar varias cámaras nun vehículo de grandes dimensións, ou a inclusión de GPS e 3G para convertelo nun dispositivo IoT e por exemplo poder seguir á bicicleta en caso de roubo.

Apéndice A

Apéndices

Apéndice B

Contido DVD

- Ficheiros fonte do código da aplicación BikeView.
- Instalador .apk da aplicación BikeView.
- Ficheiro fonte do código da aplicación LightsServer.
- Fotos do dispositivo.

Apéndice C

Requisitos e instalación

Para a instalación do software no dispositivo BikeCam neceitaremos unha Raspberry Pi con unha tarxeta micro sd cuha capacidade mínima de 4GB.

Comezarase por descargar ultima versión de raspbian lite da web da Rasberry Pi e gravar a imaxe na tarxeta micro sd, isto pódese acadar mediante o comando *dd* ou mediante unha ferramenta con ese propósito como balenaEtcher. Unha vez finalizado accederase á partición boot coa que conta agora a tarxeta de memoria, nela crearemos un arquivo baleiro chamado ssh para activar a interface ssh cando o sistema arranque, creamos tamén nesta ubicación un arquivo chamado wpa_supplicant.conf para configurar a conexón wifi, nel introducirmos os datos do SSID e a cotraseña da rede Wi-Fi creada polo dispositivo android

country=COUNTRY
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network=
ssid="SSID"
psk="PASSWORD"
key_mgmt=WPA-PSK

Expulsamos a tarxeta e a introduciomos na Raspi Pi e a encendemos, dende un ordenador ou dispositivo móbil averiguamos a direción ip, con por exemplo o softwar Fing dispoñible para linux e android. Conectamonos por ssh co usuario e contraseña por defecto "pi" "raspberry", é altamente recomendable cabiar a cotnraseña por defecto co comando passwd.

Executamos sudo raspi-config e no apartado Interfacacing Options activar o modulo da cámara, no apartado advanced options executamos a opción enlage filesistem. A continuación executamos o apartado update, de seguido poderemos saír da raspi-config e actualizar o sitema cos comandos sudo apt update e sudo apt upgrade

Para instalar a librería de jgarff para o control de los LEDs instalamos a seguinte aplicación sudo apt install scons descargamos aa librería con

wget https://github.com/jgarff/rpi_ws281x/archive/master.zip

descomprimimos con unzip master.zip entramos na carpeta e executar o comando scons executamos sudo apt install python-dev swig.

A continuación entramos na carpeta python e executamos sudo python ./setup.py build e a contunuación sudo python ./setup.py install

Apéndice D

Glosario de acrónimos

LiPo *Lithium Polymer .*

SRAM *Static Random-access Memory.*

PWM *Pulse Width Modulation.*

SPI *Serial Peripheral Interface.*

UART *Universal Asynchronous Receiver/Transmitter.*

I2C *Inter-Integrated Circuit.*

FPGA *Field Programmable Gate Arrays.*

IoT *Internet of Things.*

RISC *Reduced Instruction Set Computer.*

RGB *Red Green Blue.*

CSI *Camera Serial Interface.*

PLA *Polylactic Acid.*

ABS *Acrylonitrile Butadiene Styrene.*

OTG *On the Go.*

FPS *Frames Per Second.*

UDP *User Datagram Protocol.*

TCP *Transmission Control Protocol.*

IP *Internet Protocol.*

HTTP *HyperText Transfer Protocol.*

GPIO *General-prupouse Input/Output.*

NAL *Network Abstraction Layer.*

Apéndice E

Glosario de termos

Bytecode Código independente da máquina que xeran compiladores de determinadas linguaxes (Java, Erlang,...) e que é executado polo correspondente intérprete.

Lipo

Toolchain Conxunto de ferramentas encadeadas para desenvolver un produto de software.

Buffer Memoria utilizada temporalmente para almacenar datos de entrada ou saída durante a transmisión.

Kernel Nucleo do sistema operativo, encargado da xestión do resto de elementos do sistema.

Aliassing Efecto que causa que dúas sinais diferentes sexa indistinguibles por falta de resolución de mostraxe.

Streaming Retransmisión en directo dun contido ou sinal.

Bitstream Secuencia de bits.

Socket Interface para intercambiar un fluxo de datos nunha conexión.

Bitrate Indica o numero de bits por unidade de tempo.

Thread Secuencia de instrucións que pode ser manexada por un planificador como pode ser o sistema operativo.

STL Stereolithography, formato de arquivo para a representación tridimensional da estrutura dunha figura .

GCODE Linguaxe de programación para o control numérico.

Systemd Conxunto de daemons de administración do sistema, utilizado varios sistemas Linux.

Log Rexistro escrito dun evento.

Callback Función pasada como argumento de outra, que permite aumentar a abstración do código.

Script Programa escrito nunha linguaxe de scripting, que permite a execución de ordes nun intérprete sen necesidade de compilación previa.

Bibliografía

[1] V. G. Ruiz, "Percepion Visual," 2014. [Online]. Available: https://w3.ual.es/~vruiz/Docencia/Apuntes/Perception/Visual/index.html

[2] D. G. de Tráfico, "Ws2812b intelligent control led integrated light source." [Online]. Available: https://www.juntadeandalucia.es/export/drupaljda/Instruccion_18_S_146_luces_parpadeantes_bicicletas.pdf

[3] I. de Trànsit i Seguretat Viària FACTHUM.lab Universiat de València, "ESTUDIO: ANÁLISIS DE LA SINIESTRALIDAD EN CICLISTAS. 2008-2013," p. 110, 2016. [Online]. Available: https://www.antena3.com/a3document/2016/04/25/DOCUMENTS/01109/01109.pdf

[4] F. Mocq, *Raspberry Pi 3 o Pi Zero: explote todo el potencial de su nano-ordenador*, ser. Recursos Informáticos. Cornellá de Llobregat: ENI, 2017.

[5] D. G. de Tráfico, "Las principales cifras de la Siniestralidad de los Ciclistas. España 2016," p. 83, 2017. [Online]. Available: http://www.dgt.es/Galerias/seguridad-vial/estadisticas-e-indicadores/publicaciones/informes-monograficos/2017-2430_Las_principales_cifras_de_siniestralidad_ciclistas_2016_ACCESIBLE.pdf

[6] D. Edewaard, "The Nighttime Conspicuity Benefits of Static and Dynamic Bicycle Taillights," *All Theses*, May 2017. [Online]. Available: https://tigerprints.clemson.edu/all_theses/2620

[7] "Fly6 CE - Rear HD Bike Camera + Light." [Online]. Available: https://cycliq.com/bike-cameras/fly6ce/

[8] "HEXAGON - Camera, Signals, & Sensors for Cyclists." [Online]. Available: https://www.indiegogo.com/projects/2059770

[9] "Top-down and bottom-up design," Aug. 2019, page Version ID: 911694205. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Top-down_and_bottom-up_design&oldid=911694205

[10] "Advanced Video Coding," Aug. 2019, page Version ID: 913242779. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Advanced_Video_Coding&oldid=913242779

[11] D. Molloy, *Raspberry Pi® a fondo para desarrolladores.* Barceloina: Marcombo, 2018.

[12] J. Garff, "Userspace Raspberry Pi PWM library for WS281x LEDs: jgarff/rpi_ws281x," Aug. 2019, original-date: 2014-09-03T13:18:19Z. [Online]. Available: https://github.com/jgarff/rpi_ws281x

[13] D. Bull, "Guide to setting up LiPo batteries on the Raspberry Pi: NeonHorizon/lipopi," Aug. 2019, original-date: 2015-12-13T21:37:39Z. [Online]. Available: https://github.com/NeonHorizon/lipopi

[14] A. Leiva, *Kotlin for Android Developers: Learn Kotlin the easy way while developing an Android App.* Leanpub, 2018.

[15] "Understand the Activity Lifecycle | Desarrolladores de Android." [Online]. Available: https://developer.android.com/guide/components/activities/activity-lifecycle

[16] "Sensors." [Online]. Available: https://developer.android.com/guide/topics/sensors

[17] "Lux," Aug. 2019, page Version ID: 910867975. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Lux&oldid=910867975

[18] Consti10, "Contribute to Consti10/myMediaCodecPlayer-for-FPV development by creating an account on GitHub," Nov. 2018, original-date: 2015-12-28T11:18:13Z. [Online]. Available: https://github.com/Consti10/myMediaCodecPlayer-for-FPV

[19] "Network Abstraction Layer," Aug. 2019, page Version ID: 911058836. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Network_Abstraction_Layer&oldid=911058836

[20] "MediaCodec." [Online]. Available: https://developer.android.com/reference/android/media/MediaCodec

[21] Thingiverse.com, "Customizable Bike Mount for Smartphone by PHolzwarth." [Online]. Available: https://www.thingiverse.com/thing:3066652

[22] "Ws2812b intelligent control led integrated light source." [Online]. Available: https://cycliq.com/bike-cameras/fly6ce/

[23] J. del Estado, "Ley orgánica 3/2018, de 5 de diciembre, de protección de datos personales y garantía de los derechos digitales." [Online]. Available: https://www.boe.es/eli/es/lo/2018/12/05/3